US-9-99-002
US (Charlotte)

证　明

本证明之附件是向本局提交的下列专利申请副本

申　请　日：1999 08 11

申　请　号：99 1 11784.0

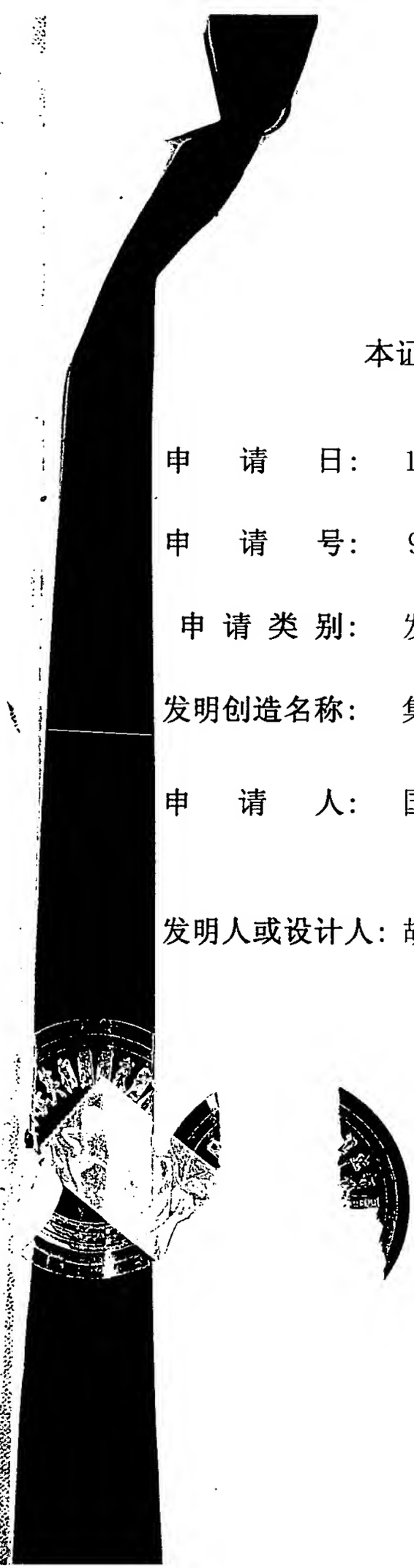
CERTIFIED COPY OF
PRIORITY DOCUMENT

申请类别：发明专利

发明创造名称：集成银行核心业务的方法和系统

申　请　人：国际商业机器中国有限公司

发明人或设计人：胡宪忠



中华人民共和国
国家知识产权局局长　姜颖

2000 年 6 月 14 日

权 利 要 求 书

1. 一种集成的银行核心业务系统，其特征在于包括：

用于提供应用业务开发环境的业务平台，在该业务平台中集成了对于各种银行核心业务来说是共同的处理部分，并为各种银行核心业务共享数据建立核心业务共享数据资料库，以便对客户情况、业务情况进行分析，为系统提供综合的客户、业务情况；

基本业务规则库和共同函数库，基本业务是能够独立地完成某种完整的业务操作的最小单位，而共同函数是实现基本业务中可能被经常重复使用的功能的程序；和

应用业务子系统，通过对基本业务规则进行组合构成各种具体的应用业务。

2. 根据权利要求1的银行核心业务系统，其特征在于所述银行核心业务包括活期存款、定期存款、贷款、结算、信用卡/储蓄卡、会计、电子汇划、清算、帐簿等业务中的至少一项业务。

3. 根据权利要求1的银行核心业务系统，其特征在于所述业务平台包括连线交易主控模块、数据库界面模块、连线批处理主控模块、连线报表模块、外部接口模块以及模拟测试模块中的至少一个模块。

4. 根据权利要求1的银行核心业务系统，其特征在于所述业务平台包括会计处理、冲正处理、24小时运行模式中的至少一个处理。

5. 根据权利要求4的银行核心业务系统，其特征在于所述会计处理是采用“会计服务器”的方式来进行会计记帐的，让各个交易或基本业务在进行记帐时，通过固定的界面来处理其会计分录产生和记帐。

6. 根据权利要求1的银行核心业务系统，其特征在于所述银行核心业务系统基于所述基本业务规则实现业务的统一连动和全面自动转帐。

7. 根据权利要求1的银行核心业务系统，其特征在于还包括用于建立系统所使用的数据格式及内存表格的生成器。

说　　明　　书

集成银行核心业务的方法和系统

本发明涉及金融交易系统，特别是涉及一种集成银行核心业务的系统和方法。

近年来，金融机构大幅度变化。目前，银行方面与以前相比能够提供更多元化的产品与服务。这些产品和服务都和计算机技术的发展密切相关。例如建立在计算机和网络基础上的金融交易卡业务、自动存取款业务、销售点自动转帐服务、电子汇款、企业银行、家庭银行、电话银行和卫星银行服务等。但这些产品与服务自成一体，重复开发，于是造成了资源的浪费，并且各产品和服务不能共享客户数据，无法为系统提供综合的客户、业务信息。

为了克服现有的银行业务系统中存在的以上问题，本发明提供一种集成银行核心业务的系统和方法，它涵盖的主要银行核心业务包括活期存款、定期存款、贷款、代理、结算、信用卡/储蓄卡、会计、电子汇划、清算、帐簿、客户信息共用等。未来将涵盖支付系统、外汇系统、投资系统等。本发明独立于各个具体业务的架构模块，同时还为开发各种新兴银行业务提供一个灵活而且功能强大的平台。

根据本发明一个方面的集成的银行核心业务系统包括：用于提供应用业务开发环境的业务平台，在该业务平台中集成了对于各种银行核心业务来说是共同的处理部分，以便对客户情况、业务情况进行分析，为系统提供综合的客户、业务情况；基本业务规则库和共同函数库，基本业务是能够独立地完成某种完成的业务操作的最小单位，而共同函数是实现基本业务中可能被经常重复使用的功能的程序；和，应用业务子系统，通过对基本业务规则进行组合构成各种具体的应用业务。

根据本发明的另一个方面的集成银行核心业务的方法包括步骤：对应用业务进行分析，建立用于提供应用业务开发环境的业务平台，

在该业务平台中集成了对于各种银行核心业务来说是共同的处理部分，并为各种银行核心业务共享数据建立核心业务共享数据资料库，以便对客户情况，业务情况进行分析，为系统提供综合的客户、业务情况；对应用业务进行分析，得出基本业务规则，建立基本业务规则库并将共同处理功能设计成共同函数库，其中基本业务是能够独立地完成某种完整的业务操作的最小单位，而共同函数是实现基本业务中可能被经常重复使用的功能的程序，和通过对基本业务规则进行组合构成各种具体的应用业务.

通过以下结合附图对本发明实施例的详细描述，可以使本发明的目的、特点和优点更加清楚.

附图说明

图1示出了根据本发明一个实施例的集成银行核心业务系统的各组成部分.

图2示出了架构控制模块的结构.

图3示出了架构控制模块与其它模块之间的关系.

图4示出了架构控制模块可以调用的各种功能.

图5示出了消息格式器的工作流程.

图6示出了应用服务器的工作流程.

图7示出了数据库界面模块.

图8示出了CBS系统整体处理模型.

图9示出了文件访问模型.

图10示出了连线批处理模型.

图11示出了连线报表的处理流程.

图12示出了预约报表处理流程.

图13示出了系统外部接口模块的架构.

图14示出了模拟测试模块的工作流程.

图15示出了测试KB/CF的工作过程.

图16示出了连线记帐处理模式.

图17示出了批处理记帐流程.

图18示出了冲正处理流程.

图19示出了次日帐切点情况.

图20示出了ATM轧帐情况.

图21示出了ATM作业模式与整批时段（Time Slot）

图22示出了24小时交易控制流程.

图23示出了KB连动处理流程.

如图1所示，本发明的集成银行核心业务的系统在设计上承接需求分析的数据模型（ERD）及处理模型（DFD）所产生的数据字典，以生成器（Generator）建立系统所使用的数据格式（Data Layout）及内存表格. 同时开发银行业务应用平台（Business Platform），以承载银行复杂的应用交易（Application Transaction），进而增进应用程序的开发及维护效率. 在应用业务上，分析出银行的基本业务处理规则，将规则设计成业务知识模块（Knowledge Block），并将共同处理功能设计成共用函数（Common Function）. 最后由应用交易（Application Transaction）组合成业务运行（Operation）系统.

以下通过业务平台，基本业务规则库和共同函数库以及应用业务子系统三个方面详细地描述本发明的集成银行核心业务系统CBS.

1. 业务平台（Business Platform）

为了提供集成的银行核心业务系统，根据本发明优选实施例，在业务平台中集成了对于各种银行业务来说是共同的处理部分，并为各种银行核心业务共享数据建立核心业务共享数据库，以便对客户情况、业务情况进行分析，为系统提供综合的客户、业务情况. 以下结合附图描述业务平台的主要模块和关键的处理流程. 其中主要模块包括：架构控制模块（CCBMain）、数据库界面模块（DBIMain）、文件界面模块（FAIMain）、连线批处理主控模块（CenterCut Main）、连线报表模块（Online Report）、外部接口模块（External Interface）、模拟测试模块（Testing Driver）关键的处理流程包括：会计处理（Accouting Process）、冲正处理（EC Process）、24小时运行模式（24 Hours Mode）、KB的连动处理.

1.1 架构控制模块

架构控制模块是CBS中连线交易的主控模块，也是整个系统的核心．图2示出了架构控制模块的结构．

在CBS系统中，与具体系统平台，如主机、操作系统、中间件、通讯协议和数据库等，相关的功能都集中在系统功能调用界面和数据库界面模块内，并将这些功能作为通过独立于具体平台产品的统一界面提供给系统内的其它模块，从而避免了系统对平台的依赖性，使得CBS系统具有很好的平台独立性和适应性．

同时CBS的大多数其它模块，尤其是架构控制模块、应用服务器模块、基本业务规则/共用函数和应用交易等，都仅使用了纯COBOL语言，而且符合ANSI 85标准，也使得CBS系统具有出色的可移植性．在移植过程中，大多数程序，特别是所有的应用业务程序，仅需重新编译就可以立即使用．

图3示出了架构控制模块与其它模块之间的关系．图4示出了架构控制模块可以调用的各种功能．

1.1.1 消息格式器

TDT (交易定义表) 表

每个应用业务子系统都各有一个TDT表．其中定义该子系统中每个交易的输入信息中的栏位、栏位的顺序、数据类型、长度、是否必输、该栏位在TFT表中的位置、转换后数据类型和长度．

TFT表

每个应用业务子系统都各有一个TFT表，它是该子系统所有交易的输入栏位的集合．各子系统的交易执行时，从TFT表中得到各自的输入信息．

消息格式器工作流程

图5示出了消息格式器的工作流程:

· 读取对应子系统的 TDT表

· 找到该交易对应的 TDT表中的记录

· 根据 TDT 记录中定义的输入栏位顺序、数据类型、长度和是否必输，依次检查各输入栏位的取值是否合法

· 根据数据类型、长度、栏位在 TFT 中的位置、转换后数据类型和长度依次转换 输入栏位并置入该子系统的 TFT表中

1.1.2 应用服务器

应用服务器完成公共的业务处理，如帐务处理、重要凭证控管、客户信息建立等等. 其具体工作情况参见图6.

1.2 数据库界面模块

本模块通过统一的服务界面 PDBIMAIN 向其它模块提供数据库访问服务. 其主要作用包括:

· 使其它模块程序不包括依赖于特定数据库形态或产品的语句. 当改变底层数据库产品时，所有应用程序不作任何变动就可以继续使用. 即实现 CBS系统对数据库的 独立性.

· 使得 CBS系统可以支持多种数据库形态和产品，如层次型数据库和关系型数据库. 进而在同一个系统中，可以同时使用多种数据库管理系统，将不同的数据库存放在不同的数据库管理系统中，从而满足特殊的业务和性能需要.

图7示出了数据库界面模块，其包括:

· PDBIMAIN 数据库界面模块服务接口
· GDBIMAIN 数据库界面模块主控程序
· GDBI4IMS IMS 数据库服务主控程序
· GDBI4RDB 关系型数据库服务主控程序
· Access Engine 数据库访问引擎.

图8示出了整体处理模型.

如图所示，GDBIMAIN处理流程为:

1) 初始化工作变量，取得所需的系统工作区

2) 输入检查，包括普通栏位检查、二级索引检查、 DB名称/分段名称/栏位名称检查等

3) 非必输参数缺省处理

4) 格式化数据库访问请求

5) 调用访问引擎存取数据库

6) 准备输出结果

7) 错误处理

下面介绍部分重要的相关系统工作区

· TDBISIDX(二级索引搜索表) 用于检查二级索引是否存在，以及得到对应的数据库名称和数据库序号

· TDBISSEG(分段搜索表) 用于数据库名称、分段名称和栏位检查，以及得到数据库的根分段名称或分段的关键字的名称

· CBAPAARE(访问路径区)记录当前的数据库访问路径。数据库访问路径由到 目前为止的本交易对该数据库的操作所决定。GDBI4RDB根据 CBAPAARE 和应用 程序本次数据库操作指令(PDBIMAIN)决定操作数据库的具体方式。

此外，在 RDBMS 环境中，对每个表的访问都通过一个特殊的程序实现，这个程序就是该表的访问引擎。所有的访问引擎都由特殊的生成器(详见生成器)自动产生。通过修改生成器的参数或扩展生成器功能可以产生具有不同功能的访问引擎，而不必修改应用程序。

另外，GDBI4RDB具有可扩展性。根据 GDBIMAIN的层次型架构和访问引擎设计，未来增加新的功能时，尤其是 RDBMS的独特功能，如连接、求和和复杂查询等，可以通过调整生成器增加访问引擎的功能，增加新的 PDBIMAIN的功能号，并在GDBI4RDB中作相应调整，即可迅速实现。

1.3 文件界面模块

本模块通过统一的服务界面 PFAIMAIN 向其它模块尤其是整批处理作业程序提供文件访问服务。文件界面模块的设计使得应用程序不依赖于具体的文件系统，保证很好的可移植性。

图9示出了文件访问模型。在 CBS 中，对每个文件的访问都通过一个特殊的程序实现，这个程序就是该文件的访 问引擎。所有的访问引擎都由特殊的生成器(详见生成器)自动产生。通过修改生成器的参

数或扩展生成器功能可以产生具有不同功能的访问引擎，而不必修改应用程序.

1.4 连线批处理主控模块

连线批处理提供用户或操作人员在批次(BATCH)的环境下利用连线交易继续批处理，尤其是批量的帐务性处理，如批量转帐等. 图10示出了连线批处理模型. 如图10所示:

1) 连线批处理可以用三种方式启动，即使用JCL、在CICS 控制台使用TCCC启动交易和在连线交易中使用GCCBAJB应用服务器等.

2) 连线批处理主控模块根据连线批处理启动时传入的不同命令调用特定的处理模块.

3) 连线批处理模块根据当前连线批处理作业的定义控制记录(Control Profile)，依次读取连线批处理输入文件中的记录，并通过CICS向架构控制模块请求执行指定的连线交易.

4) 架构控制模块根据输入的交易代码执行连线批处理作业指定的连线交易.

5) 连线交易结束时，由架构控制模块自动更新连线批处理定义控制记录，记录当前处理的输入文件记录序号，当前交易流水号，总成功笔数和总失败笔数等.

6) 架构控制模块将交易处理结果返回给连线批处理，由连线批处理处理模块写入输出文件，同时根据处理结果设置输入文件中记录的处理状态标志，以及更新连线批处理定义控制记录的相关栏位.

7) 连线批处理作业结束返回. 对于TCCC启动方式，连线批处理主控模块会根据需要将处理是否成功的信息返回至CICS控制台.

连线批处理命令是在启动连线批处理作业时，由JCL、操作员或连线交易传给连线批处理主控模块，连线批处理主控模块根据该命令完成不同的功能.

· 开始(Start) - 正常开始.

· 重启(Restart) - 可指定某一笔交易记录作为重开点.

· 暂停(Pause) - 处理设定数量交易后，暂时停止连线批处理，待一段设定时间过后，自动恢复执行原连线批处理指令.

· 修复(Recover) - 异常中断后，修复输入文件、输出文件和控制记录，为重启作准备

· 更改(Change) - 因应系统表现的变动，调校暂停的设定参数(笔数及暂停时间).

· 查询(Display) - 查询连线批处理的定义控制记录及处理状况.

· 停止(Stop) - 终止目前连线批处理. 若需继续，则须下中断后重开指令.

下面再介绍一下连线批处理的重启和修复

· 重启: 重启是指，连线批处理作业在执行过程中，被操作员人工中断或者程序异常退出， 然后重新启动该作业，并从上次中断处继续处理的过程. 重启开始处理的第一条输入文件记录可以由重启命令的参数确定. 当该参数未输时，为定义控制记录中当前处理记录序号栏位指定的输入文件记录的下一条.

· 修复: 在某些异常情况下，例如连线批处理主控模块和架构控制模块之间的链接中断等，架构控制模块成功执行连线交易后，连线批处理主控模块可能不能成功的向输出文件中插入对应的输出记录，以及更新输入文件中的处理标志栏位和控制记录中的某些栏位. 当发生这种情况时，如果直接重启该作业，由于输入文件、输出文件和控制记录没有被正确更新，可能导致后续作业发生错误.

修复就是在发生这类情况后，根据连线交易结束前架构控制模块在控制记录中记录的该记录在输入文件中的序号和交易流水号，恢复输出文件中的输出记录，以及输入文件和控制记录中相应栏位的过程. 当连线批处理作业由于Abend而中断时，必须先修复，然后才可以重启. 上次处理状态及中断原因可以从控制记录中的处理状态栏位得到.

修复工作流程

1) 从控制记录中取得上次处理的最后一笔记录对应的交易流水号

2) 根据交易流水号从重新认证数据库(BCMVOHD)中取得对应交易的输出记录. 连线交易是否在处理成功后记录BCMVOHD由交易定义数据库(BCMTXPD)中的重新 认证标志决定, 因此需要将连线批处理作业对应连线交易的交易定义数据库中的重新认证标志栏位设为Y.

3) 检查上次处理是否未将最后一笔输出记录写入输出文件中, 如果是, 则根据 BCMVOHD 记录将遗漏输出记录写入输出文件.

4) 修复输入文件和控制记录中的相应栏位, 并修改控制记录中的处理状态栏位, 标志修复已经成功完成可以重启.

下面再介绍一下连线批处理的冲正.

连线批处理的冲正为另一个连线批处理作业, 其处理流程与普通连线批处理作业相同. 它执行其对应正常连线批处理作业的连线交易的冲正交易. 它的输入文件内容对应被冲正的正常连线批处理作业的输出文件中的处理成功记录. 连线批处理模块GCCBCEC就根据被冲正作业的输出文件内容生成冲正连线批处理作业的输入文件. 其工作流程如图10所示,

1.5 连线报表模块

连线报表的功能为将端末发出的列印请求放入主机的报表产生等待队列中. 由主机的报表数据产生程序产生报表后, 通知端末启动报表下传功能, 将报表下传端末系统, 由端末系统进行列印的管理.

等待序列可分为及时及预约两种方式.

连线报表处理的基本原则为:

· 报表序号 - 6位本日唯一, 由主机自动产生.

· 等待序列分为本日与预约两种.

· 排序方法为全分行采先进先出法.

· 主机只处理原始数据, 有关报表格式, 表头、表尾、分页、合计、统计、及栏位编辑由端末系统处理.

· 主机将记录报表状态, 其状况可分为: 排序中→产生中→可下传→下传中→可列印

· 如果主机发现请求错误，将状态置为请求异常.

· 端末需执行报表列印的有关功能，例如状态查询、报表内容查询、列印、重印、份数控制、删除、安全控制、统计等功能.

图11示出了连线报表的处理流程

· 报表请求(请求格式)

· 报表请求回应(回应格式)

· 生成报表通知(连线报表)

· 连线报表下传(请求)

· 连线报表下传(回应)

· 逐笔数据

图12示出了预约报表处理流程.

· 异常处理

当端末启动查询主机的报表数据产生状态为‘可下传’但数据未传至端末，此时端末 提供下传请求交易，直接启动主机连线报表下传交易，进行报表数据下传.

· 连线报表查询连动下载请求(请求格式)

· 连线报表查询连动下载回应(回应格式)

1.6 外部接口模块

图13示出了系统外部接口模块的架构.

该架构的基本原则:

· 1/LINK，CAST，SPOT，OTHER的界面由架构控制模块的前置处理系统(外部接 口)所控制.

· 由柜台启动的对外接口交易透过应用服务器启动外部接口.

· 1/LINK，CAST，SPOT等现行系统尽量不更改其内部处理.

外部接口连接方式

1) 架构控制模块至外部接口

由AP呼叫不同的应用服务器来启动外部接口.

2) 外部接口至架构控制模块

通过CICS LINK方式启动架构控制模块，输出输入参数须符合端末设计说明书标准的输入输出格式.

1.7 模拟测试模块(Testing Driver)

模拟测试模块用于程序的单元测试. 主要原理是在批处理环境执行JCL，模拟连线交易，驱动架构控制模块、测试交易和被测程序. 使用测试案例文件设定被测试程序的输入参数.

图14示出了模拟测试模块的工作流程.

1) 准备被测试交易的输入消息. 输入消息按照标准的输入消息格式存放在测试案例文件中.

2) 按照JCL 模板准备测试案例的 JCL. JCL 中可以包括传给测试驱动程序 GCCBUTDV 的参数(Link Parm)，如机构号、柜员号、终端号、业务别、CICS系统Id和CICS程序Id等.

3) 使用 JCL 执行 GCCBUTDV. GCCBUTDV 从 JCL 和 测试案例文件中得到 Link Parm 和输入消息. 如果 Link Parm 参数不为空，其值覆盖输入消息中的对应栏位.

4) GCCBUTDV 使用 CICS LINK启动架构控制模块，并将输入消息传给架构控制模块. 输入消息的格式和从端末执行交易时完全相同.

5) 架构控制模块根据输入消息中的交易代码连动交易应用程序. 交易程序处理完成后返回架构控制模块.

6) 架构控制模块将处理结果按照标准的输出消息格式传给GCCBUTDV.

7) GCCBUTDV 收到处理结果后，格式化收到的数据，输出到标准输出设备. 通常为一个测试结果文件. 测试结果文件中除交易输出消息外还包括输入消息，并且有字符和 HEX 两种格式.

测试 KB/CF

图15示出了测试KB/CF的工作过程.

测试 KB/CF 的方法:

· 将测试案例中 KB/CF 每个参数的取值设定在测试案例文件中，并且使用 '@' 延续符将所有的参数合并为输入消息中的一个栏位，由GCCBUTDV传给架构控制模块

· 架构控制模块将包含KB/CF参数的栏位的字段存入TFT区的对应字段，然后启动测试交易

测试交易调用被测试的KBCF，并将TFT区中的该字段作为KB/CF 的参数传给 KB/CF，从而达成测试目的

1.8 会计处理

CBS 是采用“会计服务器”的方式来进行记会计帐的工作。“会计服务器”的概念是 让各个交易或KB在进行记帐时，通过固定的界面来处理其会计分录产生和记帐。在交易或KB而言就会变得很容易。从维护的角度来说，这模式也能更加体现系统的弹性。在没有详细介绍整个会计记帐的设计前，让我们把目前CCBS在记会计帐方面的特点罗列出来，在后面的描述里就可以加以引证。

· 通过应用服务器来记会计帐，而程序仅需关心直接相关的业务需求。

· 服务器为整个记帐的核心部分。当会计制度或其它相关需求发生变动时，通常只需要对会计服务器作相应的调整。而交易或KB不需要或仅需作轻微的改动。这不仅降低了开发和维护的成本，也有效的保证了系统尤其是会计处理部分的质量。

· 因为整个会计记帐模式是建立于不同的系统工作区表格中，所以会计分录的改动，如科目的合并或拆分等，基本上不需要修改程序的。这使得维护会计科目方面的工作变得简单，修改后的调试也容易许多。

· 交易或KB在记会计帐时是通过一些标准、简易和准确的方式来设定调用会计服务器的参数，这使交易或KB在设定启动会计服务器参数时所引起的错误能减到最低。

· 会计服务器在记会计帐前是会统一的检查整套会计分录的平衡性，使整个会计系统保持平衡。一旦发生错误，会计服务器会通过架

构控制模块提示错误代码和原因，使用户在最快的时间内发现系统所存在的问题，并加以纠正.

A 会计制度及架构

CBS的设计理念是要提供一个可用的、可维护的、可扩充的和完整的会计制度. 我们是通过下面的元素来实现整个会计概念的:

· 会计科目表

每一个系统所用到的科目都需要在会计科目表中设好才能够使用. 会计科目表是BCMMSCM对应的一笔记录. 记录内容包括:

科目号码: 4-2-2的三级科目架构. 三级科目的架构是指4位的一级科目，一级科目下可以细分为2位的二级科目而二级科目更可以细分为2为的三级科目以达到科、子、细目的三层架构来满足建设银行的会计需求.

科目名称: 科目的中、英文名称

科目性质: 科目是资产、负债、损益等

帐务性质: 科目是自营性或是政策性的科目

余额方向: 科目的余额是借方、贷方、收方、付方或其他的控制

使用级别: 该科目是那一个部门使用的(分行、支行等的控制)

科目状态: 该科目是目前可以使用的或是不能使用的状态

属性标志: 每一个科目表中的科目都有着一些特有的属性，是整个会计系统能够操作，科 目的属性是通过科目表中的属性标志来设定的

· 科目属性

科目表中有一系列的标志来表示该科目的属性(如缴纳存款准备金、授权科目、销 帐科目、人民币科目、外币科目、各种计提科目、摊提科目、折旧科目、不计息科目、额度科目、冲帐允许科目、表内外科目、最明细科目、外汇买卖科目、联行往来科目、汇总科目等)

· 多币种支持

一方面，系统通过多币种的会计架构来支持银行未来的业务需要；另一方面，多币种的支持也使CBS能够支持多个帐本别来区分政策性或商业性的业务。

· 多机构支持

核算中心或分行、支行、网点、营业机构

· 表内外科目

系统支持表内外科目的处理模式

· 总帐

系统支持了总帐与分户帐的基本原则，使银行可以通过总帐系统与各个业务子系统进行核对和相互监察

· 内部帐

系统支持了一套完整的内部帐处理模式，使银行能够对总帐进行必须及足够的控管及监察业务子系统的运作，增加了系统数据的可靠性。

· 挂销帐处理

系统提供了完整的挂、销帐的功能，使建设银行能够控制每一个挂销科目交易的准确性

· 内、外部清算

银行的业务涉及众多的外在金融机构进行各式各样的业务，而这些金融机构可以在国内或是遍布全世界的。面对着这么多的业务伙伴，强大的清算系统是不可缺少的。CBS的会计子系统就实现了完整的清算功能，包括分行与总行、分行与分行、分行与其他银行、分行与清算团体、分行与支行、支行与支行、支行与下属机构等的清算交易。

B. 会计处理中表的应用

在 CBS 会计记帐过程中，我们会常用到好几个的表。每一个表都有其特殊的用途。 在下面，我们就把这些表作一个简单的描述:

· TSYSSART(会计分录总表)

会计分录总表包含了每一个子系统每一个交易或KB可能产生的各种会计分录。每 一个交易都会产生一个或以上的会计分录，这些分

录的详细资料会在会计分录总表中加以设定。如果交易没有要求override的话，在会计分录总表中的就是交易需要 产生的整套会计分录了。这个总表的特色是按交易或KB来设定，由于交易可能由 多个KB 组合构成，所以我们是不能够从总表的内容来直接看到该交易所需要产生 的整套会计分录的。

· TSYSSOVT(会计分录Override表)

会计分录Override表是一个“会计科目-(记帐条件1，记帐条件2)”对应表。它使得 会计分录的记帐科目可以在交易执行时根据记帐条件的取值动态产生，而不是在会计分录总表中固定设定。它的主要作用是可以合并会计分录总表和记帐条件判定表中互相类似和仅仅科目不同的记录，便于这些表的维护。

· TSYSSCND(记帐条件判定表)

记帐条件判定表记录了每个子系统中的每个交易或者KB在不同的记帐条件取值组 合情况下，应当产生的会计分录在会计分录总表中的序号。该表用于在交易和KB 执行时，根据当时的各个记帐条件取值，确定应当产生会计分录。每一个子系统最多可定义16个记帐条件。具体的处理会在后面有关GCCBACND的描述中详细说 明。

· TSYSSAMT(金额对照表)

这个表是把每一个子系统预先设定好最多30个不同的金额收集而成的。这些金额用于在系统产生交易的会计分录表时确定每条分录的记帐金额。

· TSYSSMP(清算种类表)

这个表是记录了整个系统中目前支持的清算种类，以及各个子系统的交易或KB在 该清算种类条件下应当产生的清算分录。在后面有关产生清算会计分录的描述会有更详细的说明。

· TSYSSIAT(内部帐号序号Override表)

会计分录Override表是一个“内部帐序号-(会计科目，记帐条件1，记帐条件2)” 对应表。它使得会计分录的内部帐序号可以在交易执行

时根据会计科目和记帐条件的取值动态产生，而不是在会计分录总表中固定设定。

另外，在产生会计分录的过程中，交易、KB或各会计界面的程序都需要使用下面的系统工作区来完成记会计分录的功能:

· CBAIFARE(应用接口区)

这个区间是让交易或KB把与应用有关的数据送返架构控制模块，让架构控制模块能够进行其后续的处理。在记帐过程中，交易或KB是需要把与帐务有关的数据如交易的 参考号、交易种类(现金、转帐、交换)、帐本别(帐别+币种)、转出与转入帐号(与清算处理有关)等设在这区间里，让架构控制模块进行适当的 处理。

· CBAMTARE(金额设定表)

交易或KB进行记帐的第一步是要产生记帐所需的金额表，CBAMTARE就是用来把每一个交易或KB在起动记帐登记服务器前把该交易或KB所产生的记帐金额设定好。

· CBARAARE(会计分录表)

这是一个临时的交易会计分录表，在记帐登记服务器产生交易分录表前，记帐登记服务器是会先把会计分录总表中属于该交易或KB的会计分录拷至这个临时表中， 以便进行产生该交易或KB所需要产生的会计分录。

· CBAOAARE(会计分录Override表)

所有在会计分录总表中的会计分录都是可以通过交易的override的。Override的内容就是通过这个表在启动记帐登记服务器前设定好的。

· CBAEAARE(交易会计分录记帐表)

这个表是用来在正式记帐前暂存每一个交易应当产生的分录。会计服务器主程序就是用这个表来进行记帐处理的。每个交易最多产生24笔的会计分录。

C. 连线记帐过程

CBS系统的会计设计是让交易在日间连线时，通过会计服务器记各个会计科目的余额，并产生相关的交易流水，然后在日终批处理时通过搜流水的方式来产生各会计科目 的明细帐。在日终批处理产生会计明细帐前，批处理的作业是可以通过CentreCut方式 起动某些帐务性交易来进行批量帐务处理，如代收、代付、代扣、还贷或结息等。

C.1 会计服务器主要组成部分

· 架构控制模块会计流程控制

· 记帐金额设定应用服务器(GCCBAAMT)

· 记会计帐条件设定应用服务器(GCCBACND)

· 会计帐登记服务器(GCCBAAR)

· 会计服务器主程序(GCCBMAP)

- 架构控制模块会计流程控制

整个会计界面的控制核心是在架构控制模块中进行的。首先，当架构控制模块在收到从 端末送过来的一个交易时，架构控制模块就为记会计帐作准备。第一个动作就是把记会计帐所需的区间或表(CBAAOAAARE、CBAEAARE等)进行初始化，然后架构控制模块会起动相关的交易来进行处理。交易也可能通过起动有关的KB来进行业务的处理。在整个业务处理的过程中，如果需要产生会计分录，交易或KB会通过一系列的会计服务器来把交易或KB所产生的会计分录登记下来。当交易的处理正常结束后，架构控制模块是会把交易所产生的交易流水登录在帐务性交易流水中，然后会起动会计服务器GCCBMAP进行记帐处理。

- 记帐金额设定应用服务器(GCCBAAMT)

记帐金额设定应用服务器把交易或KB输入的各个记帐金额填入CBAMTARE中，用于在产生分录时计算每笔分录的记帐金额。该应用服务器有效地降低了因为编程人员键入的错误而错记金额的可能性。其设计概念会在后面的描述中详细介绍。

- 记会计帐条件设定应用服务器(GCCBACND)

记会计帐条件设定应用服务器根据记帐条件判定表，根据交易和KB输入的记帐条 件的具体取值，确定对应的会计分录条件号，该条件号对应于会计分录总表中的一套会计分录.

- 会计帐登记服务器(GCCBAAR)

会计帐登记服务器根据GCCBACND得到的会计分录条件号、CBAOAARE、CBAMTARE和各种清算条件取值，产生该交易或KB会计分录，并登记在CBAEAARE 中.

- 会计服务器主程序(GCCBMAP)

会计服务器主程序是整个连线记帐的最后一个过程. 交易在产生整套会计分录后返回架构控制模块. 架构控制模块把交易所产生的会计分录写入交易流水，然后起动会计服务器 主程序进行记帐. 服务器在记帐前检查该套会计分录的准确性和平衡性. 如果发现问题，则产生错误信息，交易失败返回. 否则，更新各个相关科目的余额.

图16示出了连线记帐处理模式.

1) 端末起动主机交易

端末系统把用户的交易请求通过SNA、VTAM传送到主机. 主机系统通过CICS起动架构控制模块对交易请求进行处理.

2) 启动CICS交易

当架构控制模块起动后，架构控制模块会对输入的交易信息进行必须的检查，并进行必须的预处理(如生成交易的流水号、把交易记录下来、初始化有关的会计资料表以便交易记帐)，然后起动对应的CICS交易来为输入信息进 行处理.

3) 调用KB

如果交易是不通过KB来进行业务处理，则交易就须要负责产生所有的会计分 录.

- 设定金额表(CBAMTARE)

在交易或KB处理业务个过程中，如果需要产生必须的会计分录，则交易或KB就 必须遵守会计界面的要求. 第一步就是产生会计金额表. 金额表的主要目的是要交易或KB通过标准规范的COBOL名称(如

LN-TX-AMT, SA-AMT-INT)设定到预先定义好的参数表中. 调用GCCBAAMT就是把这些设好的参数载入金额表CBAMTARE中. 每一个子系统最多能够定义30个不同的金额. 这个金额表在后面的处理中是需要用到的.

- 设定记帐条件

当记帐用的金额表设定好以后, 下一步的工作就是设定记帐条件. 交易或KB通过调用记会计帐条件设定应用服务器(GCCBACND)设定记帐条件. 在调用应用 服务器之前, 交易或KB需要先把判断记帐条件的各个要素的取值, 即记帐条件 , 设定好. GCCBACND就是利用这些参数在记帐条件判定表中寻找与之匹配的记录. 比对的关系可以是大于(>)、大于等于(≥)、小于(<)、小于等于(≤) 、等于(=)、不等于(≠)或其他.

记帐条件是具有业务含义的某些数据项, 例如存款种类、交易类别、贷款种类、存期等到.

- 产生和登记会计分录

在设定好记帐的条件后, 最后的工作就是产生和登记该交易或KB所需要产生的 会计分录. KB只产生与其自身业务相关的会计分录; 交易的会计分录是其调用 的所有KB和交易自己产生的会计分录的总和. 例如转帐交易是通过分别调用活 期支取KB和活期存入KB来实现转帐交易的. 那么, 转帐的整套会计分录就是这 两个KB分别产生. 每一个KB都只产生一半的会计分录, 但在交易的角度而言, 整套的会计分录是完整的和平衡的.

在调用会计帐登记服务器(GCCBAAR)前, 交易或KB需要先设定好调用的参数如 帐本别、币别、交易参考号、交易种类、转入/转出帐号等. 另外, 如果交易或 KB需要override会计分录总表中的任何一个栏位时, 交易或KB可以通过填写CBAOAARE(Override表)来覆盖会计分录总表中某一个栏位的内容. 交易甚至可以使 用CBAOAARE增加新的会计分录(即在会计分录总表中没有预先设定的会计分录). 唯一的限制就是整个交易所产生的会计分录不能超过24笔.

当所需的参数或表设定好以后，交易或KB就可以调用会计帐登记服务器(GCCBAAR). GCCBAAR先根据GCCBACND等到的记帐条件号把会计分录总表中对应的那套会计分录拷到临时会计分录表中，然后与CBAOAARE、CBAMTARE等表合并起来产生交易会计分录记帐表(CBAEAARE). 这个记帐表是当交易结束回到 架构控制模块后用来记帐用的. GCCBAAR还会产生与清算有关的会计分录，清算会计分录是会包括在24笔内的. 清算的处理模式请参考后面 有关清算的描述.

4) KB返回到交易

当KB在完成它的处理并登记了有关的会计分录后，KB就会结束并返回到调用它的交易.

5) 交易返回到架构控制模块

当交易结束后，交易就会返回到架构控制模块，让架构控制模块进行交易后的处理(如记 帐、Rollback、交易输出等).

6) 产生交易流水

如果交易是成功的话，则架构控制模块便会更新有关的交易流水.

7) 记会计帐

如果交易是会产生会计分录的话，架构控制模块 便会起动会计服务器主程序来进行真正的记帐动作. 记帐包括整套会计分录的平衡检查、记总帐、挂销帐和记内部帐.

D. 批处理记帐流程

图17示出了批处理记帐流程.

当日间的处理结束，系统便会进入晚上批处理时段. 晚上的主要处理是记总帐明细 档.

在记总帐明细档之前，每一个子系统都需要把当天在连线批处理时需要产生的会计分 录都登记在交易流水中. 主要的连线批处理作业包括代收、代付、代扣、还贷、结息等.

当所有当天的会计分录都产生以后，系统会从交易流水档中产生大传票档．这个大传票档用来产生个总帐科目和内部帐的明细记录的．产生明细记录的流程可以参照下图．

E. 清算处理

在一个银行的运作过程中，与外界银行或金融机构有着业务的往来是非常普遍的事情 ．每一笔与外界银行的交易都需要经过严格的跟踪，同时也需要与外界银行产生相对 的财务关系．这财务关系便是“清算”活动．

因为银行业务的不同或对清算概念的差异，清算模式有很多的种类(复杂或简单的)． 在CCBS的设计里，所有的清算都是以核算中心(分行)为主．支行与支行的清算是透过分行来进行；支行与外界银行的清算也是透过分行来进行．换句话说，分行就变成清算的核心了．

清算的过程是分为两个部分进行处理的．第一步是数据的收集，第二步是产生清算的 轧差，从而达到清算的目的．

在CCBS里，我们是通过记清算会计分录的方式来收集清算的数据．每一笔涉及两个不同机构的交易都需要产生对应的会计分录来反映清算活动的发生，并为日中的清算处 理产生数据．清算用的会计分录是透过“联行往来”这个科目来记录的．

我们在主机系统中建成一个清算种类表TSYSSSMP，其中包括每种清算种类会产生的作为清算用途的整套会计分录，作为产生跟清算有关的会计分录之用．

首先，在交易或KB调用GCCBAAR之前，交易或KB是需要把交易所牵涉到清算活动的支行通过AIF-TR-IN-ACCOUNT或AIF-TR-OUT-ACCOUNT设好．在GCCBAAR的处理 中，GCCBAAR会从上面的两个帐号和交易行的行号进行比较．如果这三个行号是一 样的话，则该交易就没有任何的清算活动．如果这三个行号不完全是一样的话，则会发生不同形式的清算动作．GCCBAAR就会按判断出来的清算种类产生对应的会计分录．这个过程是完全自动而不需要交易或KB做什么的处理的．GCCBAAR所产生与清算活动有关的会计分录是

会记在交易会计分录表中。当交易结束返回到架构控制模块后，真正的会计分录就会正式产生。

在晚上批处理时，日间记帐过程中所收集到有关清算活动的数据是会被用来产生各式各样的清算报表的。这些清算报表是会与外界金融机构结算时作为参考比对的。

1.9 冲正处理(EC Process)

“冲正处理”指当日冲正处理。所谓冲正处理，是指由于柜员误操作或其它 原因错误地执行了某笔交易，事后用执行一笔反向交易的方式取消原交易的错误效果 的过程。

A. 冲正的主要工作

冲正须要完成如下工作:

· 取消被冲正交易对应用数据库产生的错误效果

· 取消被冲正交易产生的会计分录对会计帐的错误更新

· 产生适当的凭证

· 其它与具体业务有关的处理

B. 冲正处理

B.1 架构控制模块和交易/KB

冲正处理由 架构控制模块和交易/KB谐调完成。架构控制模块根据被冲正交易的交易流水为冲正交易恢复有关被冲正交易的信息，如交易请求、KB请求、TGS区和会计分录的内容等，同时架构控制模块在冲正交易处理成功后为其进行与被冲正交易反向的会计处理。交易/KB根据架构控制模块提供的有关被冲正交易的信息恢复应用数据库和完成其它与具体业务相关的处理。

正常交易和冲正交易使用的同一个交易或KB程序。交易和KB中应包括正常和冲正两者处理逻辑。

B.2 相关系统工作区

· CBECIARE(冲正输入信息)

记录冲正交易的部分输入信息，如授权主管号、帐号、金额1、金额2、交易流水号等，也包括部分被冲正交易的交易流水记录中的信

息，例如：交易营业日、系统日期和系统时间等．架构控制模块 根据冲正交易输入信息填写 CBEICARE，供后续冲正 处理使用．

· 交易请求

交易请求由交易填写，交易结束时由架构控制模块记入交易流水．冲正时，冲正交易可以通过架构控制模块得到被冲正交易的交易请求的内容，用于恢复应用数据库和其它处理．交易请求的作用类似与KB请求．

图18示出了冲正处理流程．

1) 端末使用统一的冲正交易代号 CM0999800向 架构控制模块提交冲正交易请求，同时 上传被冲正交易的交易流水号、帐号、金额1和金额2等数据．帐号、金额1和金额2由柜员输入，用于和被冲正交易记录在交易流水中的对应数据进行对比．只有当两者相同是冲正才被允许进行．这种检查可以减小由于误输交易流水号而将其它交易冲正的可能性．

2) 架构控制模块根据端末传入的交易流水号，取得被冲正交易的交易流水，根据其内容 填写 CBECIARE，恢复输入消息交易输入信息、CBAEAARE会计分录、交易请求、CBTGSARE和 KB请求等至被冲正交易结束时的内容．然 后，检查输入的帐号、金额1和金额2是否和被冲正交易的交易流水中的对应栏位内容是否相同．最后，用输入消息中的内容，即被冲正交易的输入信息，启动交易．

3) 交易根据输入信息和交易请求中的内容进行冲正处理．必要时调用 KB 对正常交易中的 KB 调用进行冲正． KB 的输入参数由交易的冲正处理逻辑产生， 而不是像交易的输入信息根据交易流水恢复至与被冲正交易完全相同，所有存在 KB 在冲正时输入参数与在被冲正交易时不同的可能性．

4) 当交易通过GSYSTRIG连动KB时，GSYSTRIG 根据被冲正交易记录的CBTGSARE 中的信息找到被连动KB在被冲正交易中记录的KB请求．然后，连动KB，并将KB请求传递给被连动KB．

5) 交易和 KB 的冲正处理结束后返回架构控制模块，架构控制模块根据CBAEAARE中的内容，即被冲正交易产生的会计分录，进行帐务处理。冲正交易的会计分录的区别仅是记帐金额为原记帐金额的负值。

6) 架构控制模块 将冲正交易的输出表等回送端末。

1.10 24小时运行模式(24 Hours Mode)

针对24小时不停业务运转交易所设计的处理原则，其业务范围包含ATM/POS，Voice/-Firm Banking，Home Banking等。

时段切割

参考时间	时段切割	工作描述
6:30 pm	T1	整批作业开始 前段备份作业 主档下载
8:00 pm	T2	本日工作结束 序时资料下载 整合数据合并
10:00 pm	Tn	次日作业开始 中段备份作业 ATM/语音追帐
12:00 pm	Tc	次日转帐开始 转入作业 转出作业 后段备份作业
0:00 am~6:30 pm	T0	连线作业开始

时段切割主要目的在于JOB设计时各应用系统依其情况将其作业放入适当的时段. 同时在24小时运作模式下执行次日帐切点. 时段切点将记录于系统的CMICTDAT控制档中所有的交易知目前在何段中, 而各时段的时间为逻辑切点, 不因实际时间而影响, 营业记帐日期.

图19示出了次日帐切点情况.

1) 24小时交易换日切点置于最早的网点结帐前.

2) 采全分行统一切换.

3) 切换之后所有交易所发生的帐务以次日帐处理.

4) 未切换之前网点不能关帐.

图20示出了ATM轧帐情况.

1) 会计帐务以换日切点24小时为循环记录帐务日的每一笔交易.

2) 换钞现金以换钞切点(每一部机器不同)24小时循环记录每一笔现金交易.

图21示出了ATM作业模式与整批时段（Time Slot）

图22示出了交易控制流程.

1) 在下午支行网点结帐前由中心执行Cut off (切换)次日.

2) 架构控制模块接到24^{hrs}Tx消息(TXP定义)，将ATM模态传给24^{hrs}Tx.

3) 该交易判断连线或离线后更新主1 DB or Temp DB.

4) 在TN时启动Catch up (追帐)程序，读取TEMPFILE的消息传入架构控制模块的 24^{hrs}Tx.

5) 连续批处理作业可根据CMICT24C的设定的执行该区段，由此设定可启动多个Center Cut同时执行追帐，缩短追帐时间.

1.11 KB连动处理

图23示出了KB 连动处理流程

正常交易

1) 端末向架构控制模块提出交易请求

2) 架构控制模块为CBKBRARE、CBTGSARE和其它系统工作区分配空间，然后根据端末提供的交易代码调用被请求的交易

3) 交易调用 GSYSTRIG 连动 KB

4) GSYSTRIG 检查连动的 KB 的个数是否已经为6个. 如果是，则交易失败，输出出错信息；否则，取得 CBKBRARE 中下一个KB请求，并将 APA-KBA-ADDR 设为该 KB请求的地址. 然后，在CBTGSARE 中记录被连动 KB 的名称和连动该KB的程序名称，并将已连动 KB个数加1. 最后，调用 KB. 在某些子系统内 部，KB之间可能需要使用TFT区传递连动信息，当这些子系统的 KB 被其它子系统连动时，GSYSTRIG 将为其分配其原来所属子系统的TFT区.

5) KB通过APA-KBA-ADDR取得KB请求，并将需要记录在交易流水中的信息写入正常KB请求中. KB处理完成后返回 GSYSTRIG.

6) GSYSTRIG返回交易．如果 4)中曾经分配TFT区，则先释放该区域，然后返回．

7) 交易处理结束后返回架构控制模块．由架构控制模块将正常KB请求和CBTGSARE中的内容写入交易流水．架构控制模块退出时，释放CBKBRARE、CBTGSARE和其它系统工作区．

8) 架构控制模块将交易输出表格或出错信息回送到端末．

冲正交易

1) 端末向架构控制模块提出交易请求

2) 架构控制模块为CBKBRARE、CBTGSARE和其它系统工作区分配空间．然后架构控制模块取得原正常交易记录的交易流水，并根据交易流水的内容，恢复CBKBRARE和CBTGSARE至正常交易结束时的状态．最后根据端末提供的交易代码调用被请求的交易

3) 交易执行其冲正逻辑，并调用GSYSTRIG连动KB，对KB在原正常交易中进行的业务处理进行冲正．

4) GSYSTRIG根据CBTGSARE中的内容，以及目前被连动KB的名称和连动该KB的程序名称，找到该KB在正常交易中使用的KB请求，并且置CBTGSARE中对应该KB的记录中的是否冲正标志．GSYSTRIG 将 APA-KBA-ADDR 设为该KB请求的地址．然后，调用KB．在某些子系统内部， KB 之间可能需要使用 TFT区传递连动信息，当这些子系统的KB被其它子系统连动时，GSYSTRIG 将为其分配其原来所属子系统的TFT区．

5) KB 执行其冲正处理逻辑．在执行过程中，通过APA-KBA-ADDR 取得KB请求．正常KB请求中的内容为KB在正常交易中记录的内容．KB可以利用其中的信息进行冲正．同时，KB也可以将需要记录在交易流水中的信息写入反向KB请求中．KB处理完成后返回GSYSTRIG.

6) GSYSTRIG返回交易．如果 4)中曾经分配TFT区，则先释放该区域，然后返回．

7) 交易处理结束后返回架构控制模块。由架构控制模块将反向KB请求和CBTGSARE中的内容写入交易流水。架构控制模块退出时，释放CBKBRARE、CBTGSARE和其它系统工作区。

8) 架构控制模块将交易输出表或出错信息回送到端末。

2 业务规则(Knowledge Block)和共用函数(Common Function)

以下描述业务规则和共用函数的概念。

A. 业务规则的定义

业务规则(KB)是能够独立地完成某种完整的业务操作的最小单位。KB是构成 应用子系统的基本单元。通常，KB 具有如下特性:

- 独立完成完整的基本业务动作
- KB 的业务处理通常是帐务性的
- 更新应用业务数据库
- 包含冲正处理逻辑
- 需要记录交易流水
- 输出表

上述各项中，第一条是最重要的也是最基本的，也是 KB 区别于 CF 的最主要特征。

根据 CBS 系统的设计观念，所有的业务，即交易和交易的功能，是由某些基本的业务 规则，或业务“知识”(Knowledge)构成。各种业务就是这些“知识”的不同组合。而这些基本的业务规则就是由 KB 实现的。一方面，交易可以用 KB 进行简单的组合而构成，KB 确定了其业务规则；另一方面，交易可以通过覆盖(Override) KB 输出的表、增加与其业务相关的新的处理逻辑等方法调整交易的功能，适应具体的业务需求。

基本业务规则必须在连线环境中运行。

B. 共用函数的定义

共用函数，即共用函数，是一组实现可能被经常重复使用的功能的程序，这些功能主要包括各种计算、数据查询/维护、数据检查等。共用函数的设计可以简化应用开发和提高应用的可维护性。

共用函数，与基本业务规则相比，有如下特点：

· CF 实现的功能不能独立成为完整的业务处理

· 不能进行帐务性处理

· 不能记交易流水

· 不能冲正

· 不输出表，只通过输入/输出参数传递数据

· CF 如果不操作数据库，则也可以在批处理环境中使用

· CF 中的应用逻辑错误不直接导致交易失败。具体的错误处理由调用该 CF 的程序决定

C. KB 的连动处理

KB 是通过被交易或其他 KB 连动而运行的。交易 和 KB 都必须通过 GSYSTRIG 连动 KB.

KB请求是一组内存区域。KB 在执行过程中可以将需要记录交易流水的信息存放在KB 请求中，交易成功后架构控制模块负责将KB请求中的内容记入该交易的流水中。在交易冲正时，KB可以通过架构控制模块得到原来的KB请求内容，用于冲正处理。所以KB请求的主要作用包括：

· 为 KB 提供记录交易流水的手段。交易流水中的内容将会在批处理作业中时被 使用。

· 记录 KB 在正常执行中的某些与当时特定状态相关的中间结果，为可能的冲正处理 保留足够的信息。

一个交易在执行过程中最多可以连动 6 个 KB，其中包括 KB 连动 KB。每个 KB 请求为256 Byte.

重要的系统工作区

· CBKBRARE(KB请求区)

包括全部 6 个正常KB请求、6 个反向KB请求和KB索引. KB索引为当前 KB 使用的KB请求在CBKBRARE中的序号. 在正常 交易中, 正常KB请求的内容将被记入交易流水; 在冲正交易中, 正常KB 请求中为KB在正常交易中写入KB请求的内容, 反向KB请求中的内容将在冲正 成功后写入交易流水.

· CBTGSARE(Trigger Sequence Area)

在正常交易中, GSYSTRIG 在 CBTGSARE 中记录每个被连动 KB 的名称、连动该 KB 的程序的名称、以及目前已经连动的 KB 的个数. CBTGSARE 在交易成功后, 由 架构控制模块 写入交易流水; 在冲正交易中, CBTGSARE 中返回 GSYSTRIG 在正 常执行中记录的内容, 即正常交易连动的 KB 个数、KB 名称和连动 KB 的程序名 称等, 以及在冲正过程中当前该KB是否已经被冲正. GSYSTRIG 根据 CBTGSARE 中的内容在 CBKBRARE 中找到 KB 在正常交易中使用的 KB 请求, 并将其传递给该 KB.

· APA-KBA-ADDR

CBAPALST中的一个栏位, 指向当前 KB 使用的 KB 请求的指针, 用于向 KB 传递 KB 请求.

3 应用系统(Application System)

在 CBS 系统中, 银行应用业务主要通过各子系统的连线交易和批处理作业实现.

4 生成器(Generator)

生成器是CBS系统提供的一组工具程序, 用于自动生成部分源代码、系统表和报表等. 目前, 生成器主要用于开发、维护和批处理报表. 生成器的设计显著地减少了应用开发的工作量, 提高了应用质量.

生成器工作原理

A. 以数据字典为输入源的生成器

大多数生成器, 包括所有的数据格式生成器和部分架构表生成器, 都以数据字典为输 入源. 这样的设计保证了产生器产生的 Copybook 和架构表与前期设计和系统其它部分的一致性性.

B. 会计处理表生成器

所有会计处理表生成器均以会计表格数据库为输入源. 会计表格数据库中的数据由用户使用会计表格维护交易输入和维护. 如图26B所示.

C. 数据库访问引擎和 COPYBOOK 生成器

对于 RDBMS, 所有的Table Copybooks、 Host Structure Copybooks和访问引擎程序均 由生成器自动产生.

Table Copybooks 和 Host Structure Copybooks通常由数据字典(DDD)生成, 保证与设计 的一致性, 但是生成器也允许从IMS Segment Copybooks直接生成 RDBMS Copybook , 作为一种快速和应急途径.

生成控制表控制访问引擎生成结果, 允许根据每个数据库或Table的特性生成具有不同功能的访问引擎

D. 文件访问引擎生成器

文件属性定义文件中定义了每个文件的特殊属性, 包括 文件名称、文件类型、允许访问方式、记录最小长度、最大长度、记录是否变长、数 字栏位存放方式等等.

文件引擎生成器(FAI Generator)根据每个文件的属性定义内容生成其各自的访问引擎.

E. 报表生成器

批报表在系统开发中占有一定程度的工作量, 它的特点是数量多, 但一般来说般来说 功能上并不复杂. 从过程上来看, 要生成一个批报表可分为读取数据库、排序、产生 报表三个阶段. 传统做法一法一般是以个别的批处理功能(或含JCL)产生个别报表. 其中除读取数据库及排序因功能需求各异, 仍需个别处理外, 最后的产生报表可用一标 准化过程处理, 可省却大量在批基于上述考虑, 故将批报表的流程程固定为:

a)读取数据库产生生报表数据档

b)报表数据档排序

c)REPORT GT GENERATOR生成报表R生成报表其中REPORT GENERATOR的来源依据除报表数据档外，尚需一个参数.

5 程序模板(Skeleton)

CBS 系统为主要的应用程序种类，如 KB、CF、各种交易程序、各种批处理程序、使用各种批处理工具的JCL程序、连线批处理JCL等等，提供程序模板(Skeleton). 程序模板(Skeleton)中包含该类程序的共同处理，并定义了程序结构.

开发人员在编写应用程序时仅需要在对应的模板(Skeleton)内填入具体的功能逻辑即 可.

程序模板(Skeleton)的使用使得开发出的应用程序规范化，提高了系统质量. 同时，因为模板已经包括了所有的共同处理或规定了共同处理的格式，如初始化和错误处理等 ，开发人员只需要了解和实现具体的功能逻辑，从而减少了对开发人员在不必要技术 领域的要求，显著地提高了开发效率.

CBS 系统为主要的应用程序种类，如 KB、CF、各种交易程序、各种批处理程序、使 用各种批处理工具的JCL程序、连线批处理JCL等等，提供程序模板(Skeleton). 程序模板(Skeleton)中包含该类程序的共同处理，并定义了程序结构.

程序模板的内容主要包括:

- 程序段落结构
- 共同的处理和数据说明，如内存区初始化和 Linkage Section 的数据定义
- 错误处理方式
- 常用处理的示例

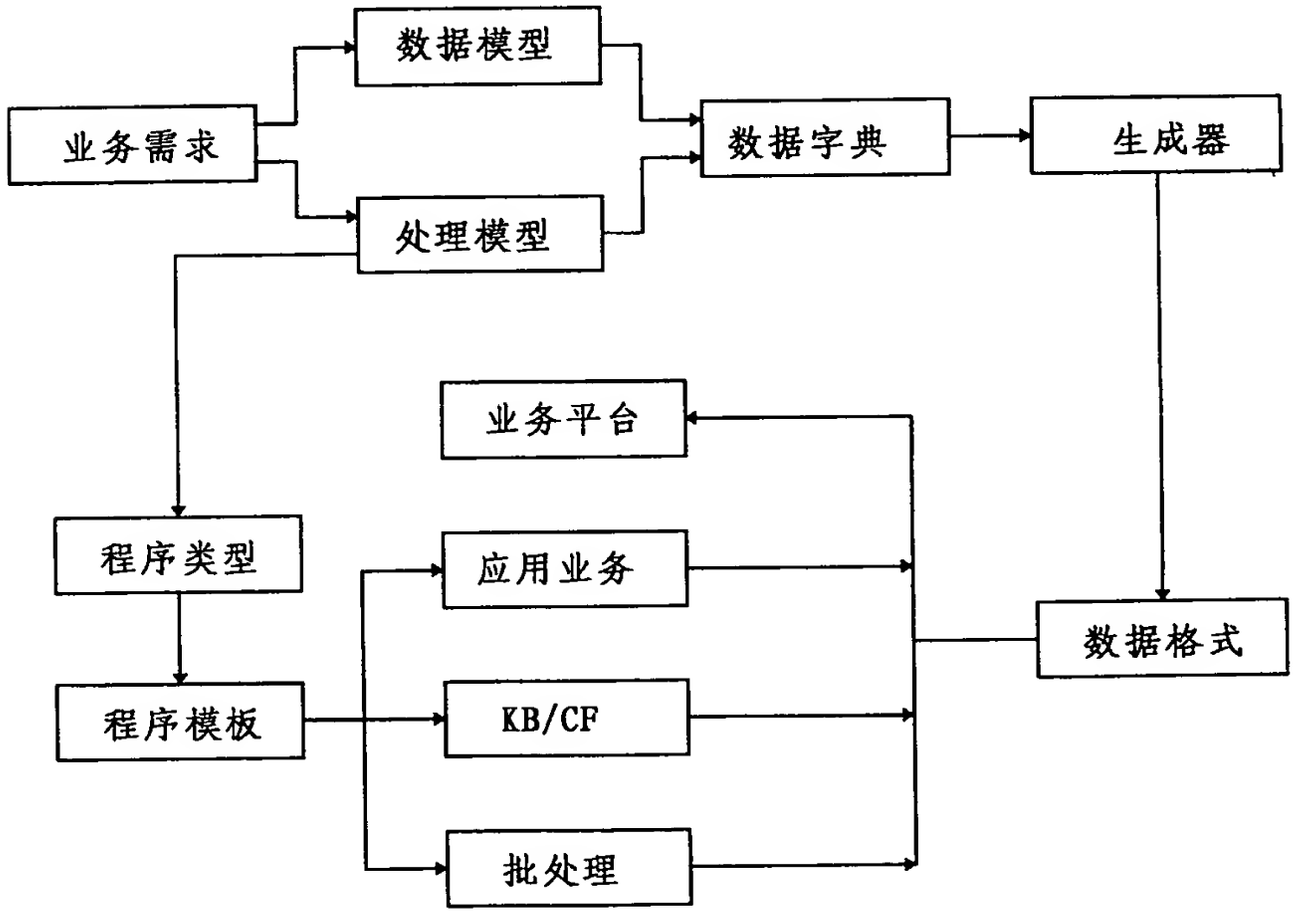
开发人员在编写应用程序时仅需要在对应的模板(Skeleton)内填入具体的功能逻辑即可. 不需要了解业务平台(Business Platform)内部处理方式和其他与硬件/软件平台相关的内容，只需要了解和实现具体的业务功能即可.

程序模板的使用使得开发出的应用程序规范化，提高了系统质量，减少了对开发人员在不必要的技术领域方面的要求，显著地提高了开发效率。

以上结合具体实施例从业务平台、基本业务规则和共用函数以及应用子系统三个方面详细地描述了本发明。本领域技术人员应清楚，在不背离本发明的精神和实质的前提下，本发明具有许多实施方式，以上实施例只为了说明本发明，不应构成对本发明的限制、本发明的保护范围由所附权利要求书来限定。

说　明　书　附　图

图1



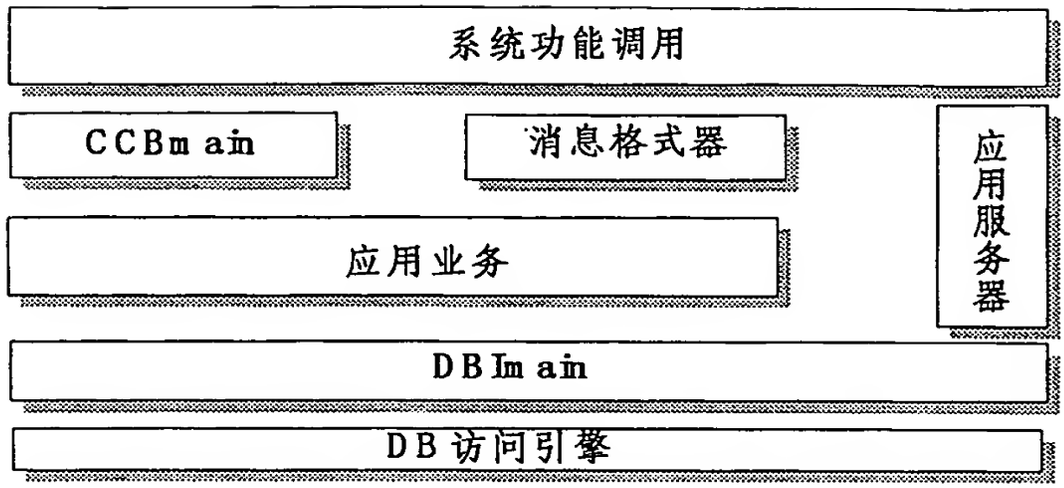

系统功能调用
CCBmain
消息格式器
应用服务器
应用业务
DBImain
DB访问引擎

图2

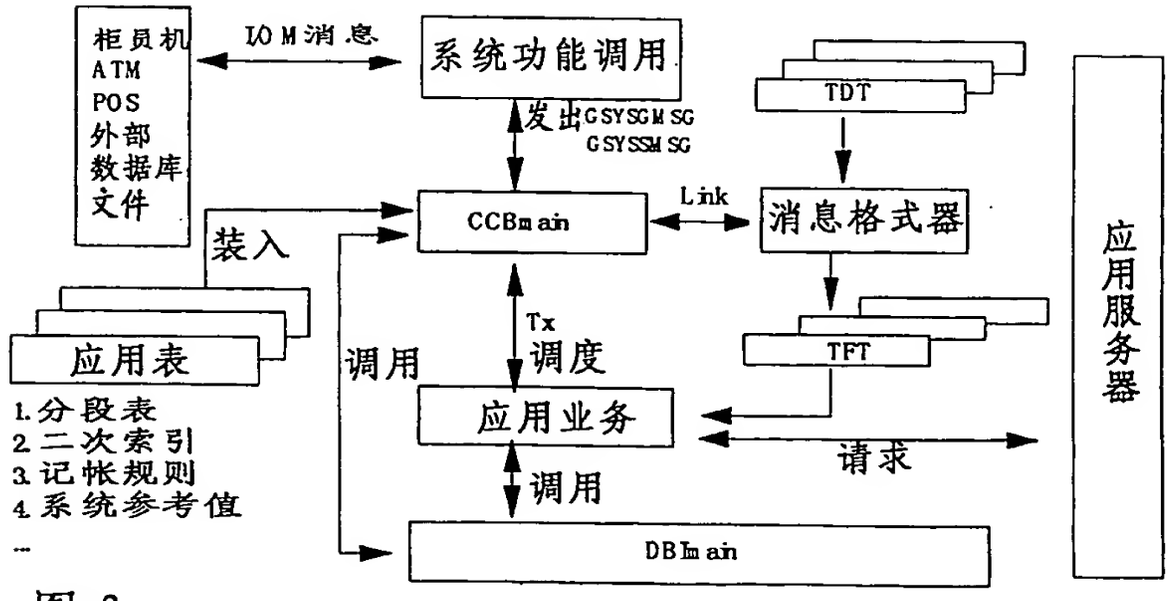

柜员机
ATM
POS
外部
数据库
文件
IOM消息
系统功能调用
发出GSYSGMSG
GSYSSMSG
TDT
CCBmain
Link
消息格式器
装入
应用表
1.分段表
2.二次索引
3.记帐规则
4.系统参考值
...
Tx
调度
调用
TFT
应用业务
请求
调用
DBImain
应用服务器

图 3

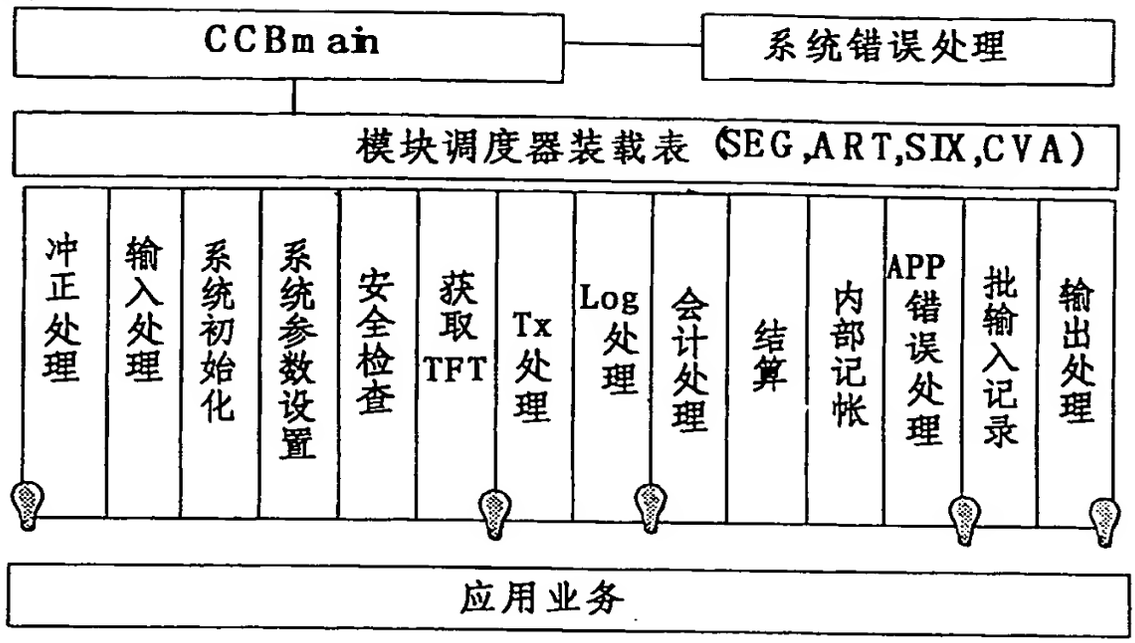

CCBmain
系统错误处理
模块调度器装载表 (SEG,ART,SIX,CVA)
冲正处理
输入处理
系统初始化
系统参数设置
安全检查
获取TFT
Tx处理
Log处理
会计处理
结算
内部记帐
APP错误处理
批输入记录
输出处理
应用业务

图4

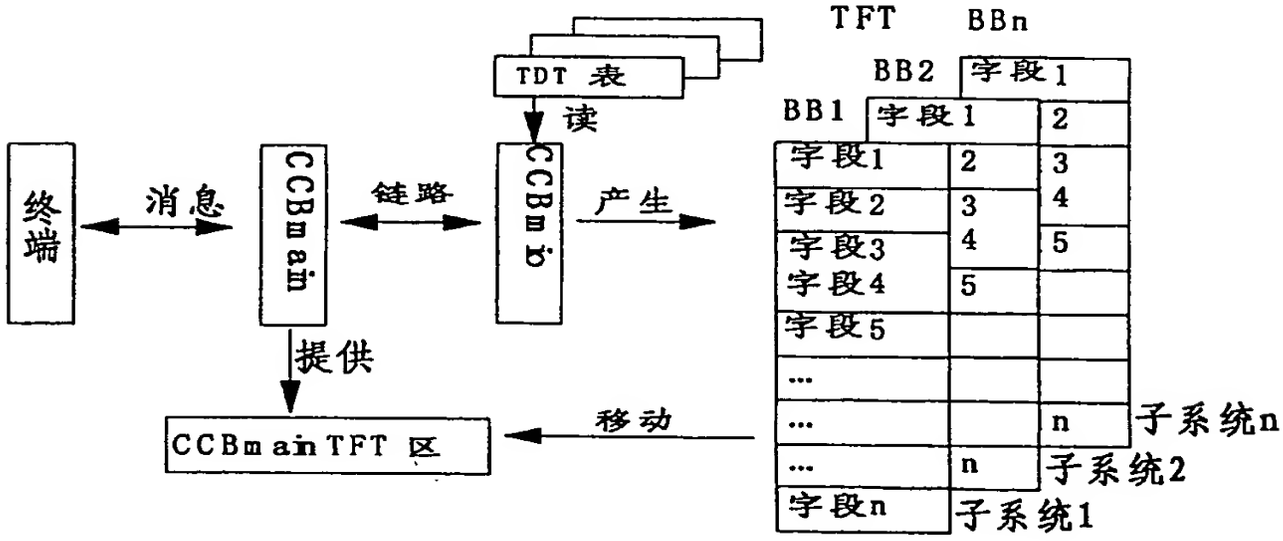
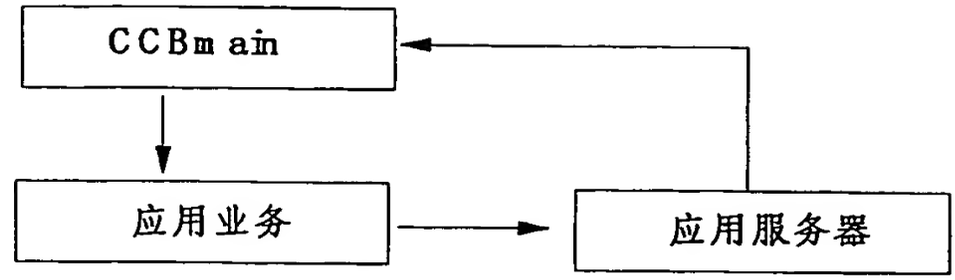

TFT
BBn
TDT 表
读
BB2
字段1
BB1
字段1
2
终端
消息
CCBmain
链路
CCBmib
产生
字段1
2
3
字段2
3
4
字段3
4
5
字段4
5
字段5
...
提供
移动
...
n
子系统n
CCBmain TFT 区
...
n
子系统2
字段n
子系统1

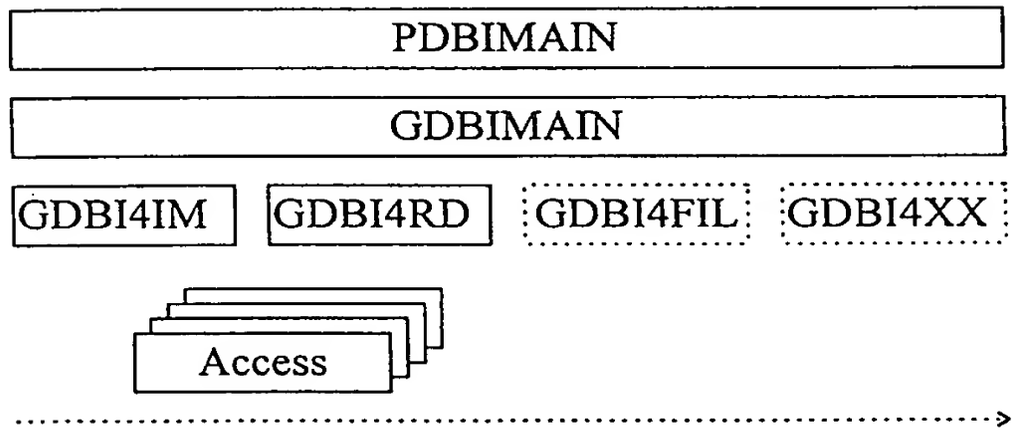
图5



GCCBAAR -AccountingRule
GCCBAVD -ValuedDocument
GCCBAMH -MessageHandling
GCCBAMP -MultipleOutputPut
GCCBACI-CustomerInformationBuild
GCCBAMT -TxAmountSeting
GCCBCND -TxAccountingConditionSeting
GCCBASD -SpecialDeviceHandling
...

A应用服务器
1.由应用连动
2.更新核心层数据库
3.向CCBmain传数据
4.由CCBmain定义参数区

图6



IMS, UDB/6000, DB2/400, RDBMS, FILE

图 7

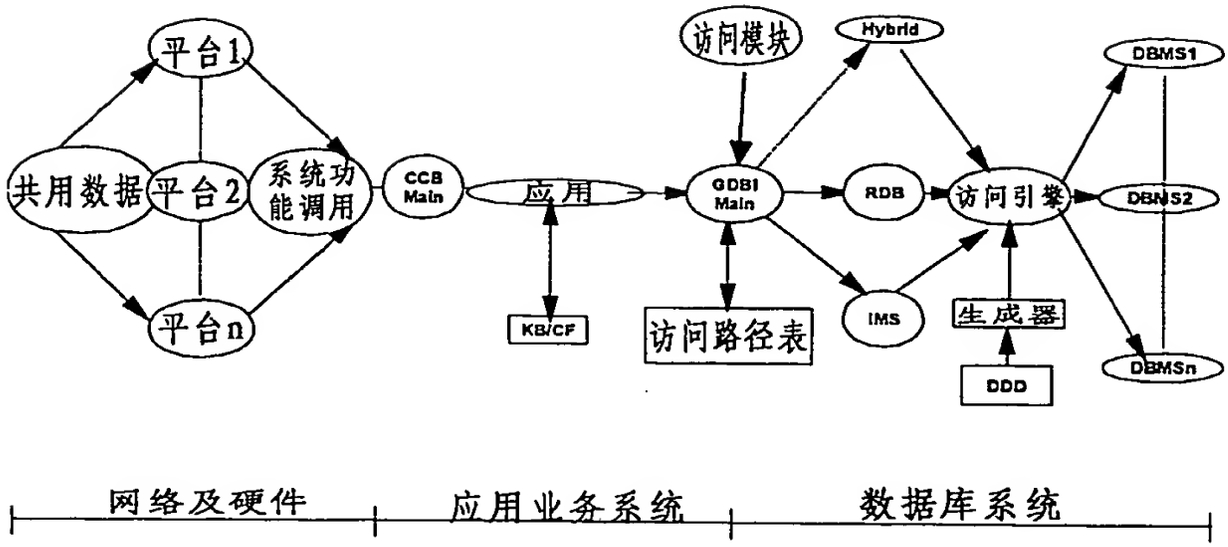

平台1
共用数据
平台2
系统功能调用
平台n
CCB Main
应用
KB/CF
访问模块
GDBI Main
访问路径表
Hybrid
RDB
IMS
访问引擎
生成器
DDD
DBMS1
DBMS2
DBMSn
网络及硬件
应用业务系统
数据库系统

图 8

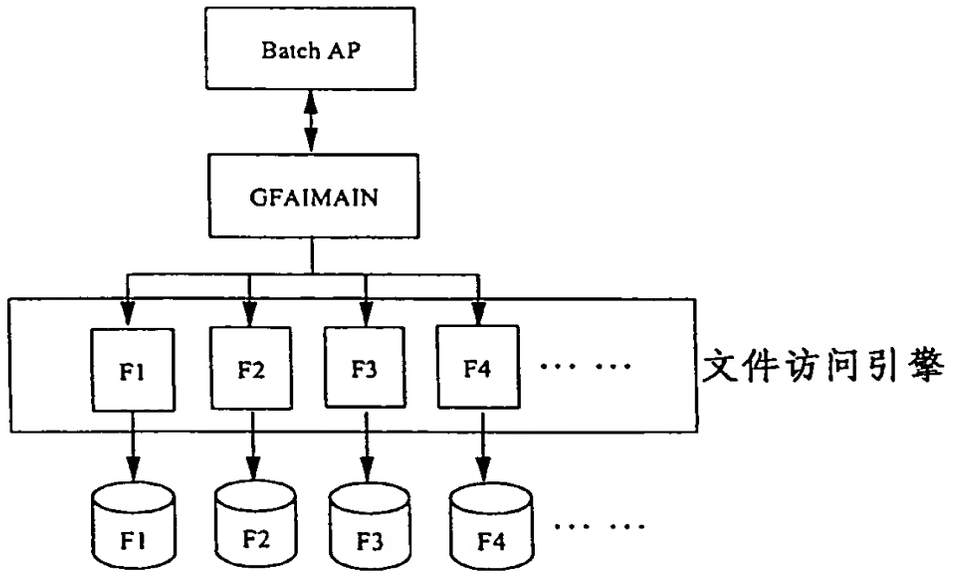

Batch AP
GFAIMAIN
F1
F2
F3
F4
文件访问引擎
F1
F2
F3
F4

图 9

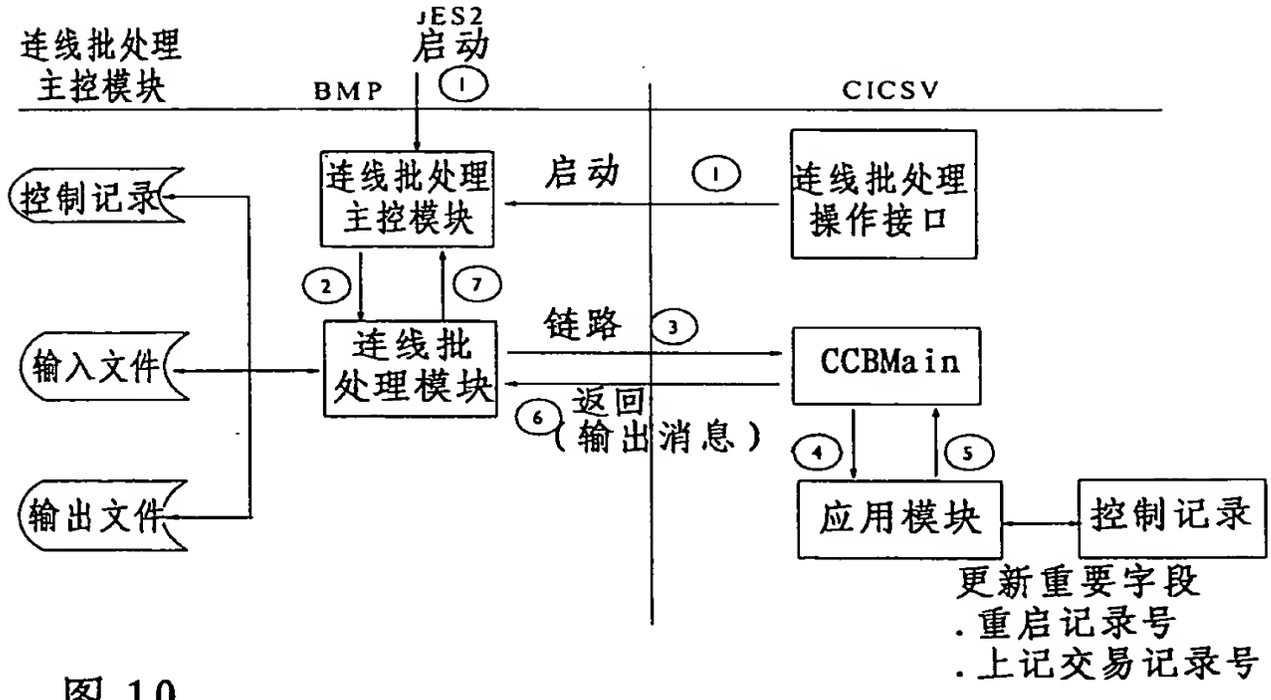

连线批处理
主控模块
JES2
启动
①
BMP
CICSV
控制记录
连线批处理
主控模块
启动
①
连线批处理
操作接口
②
⑦
输入文件
连线批
处理模块
链路
③
CCBMain
⑥
返回
(输出消息)
④
⑤
输出文件
应用模块
控制记录
更新重要字段
.重启记录号
.上记交易记录号

图 10

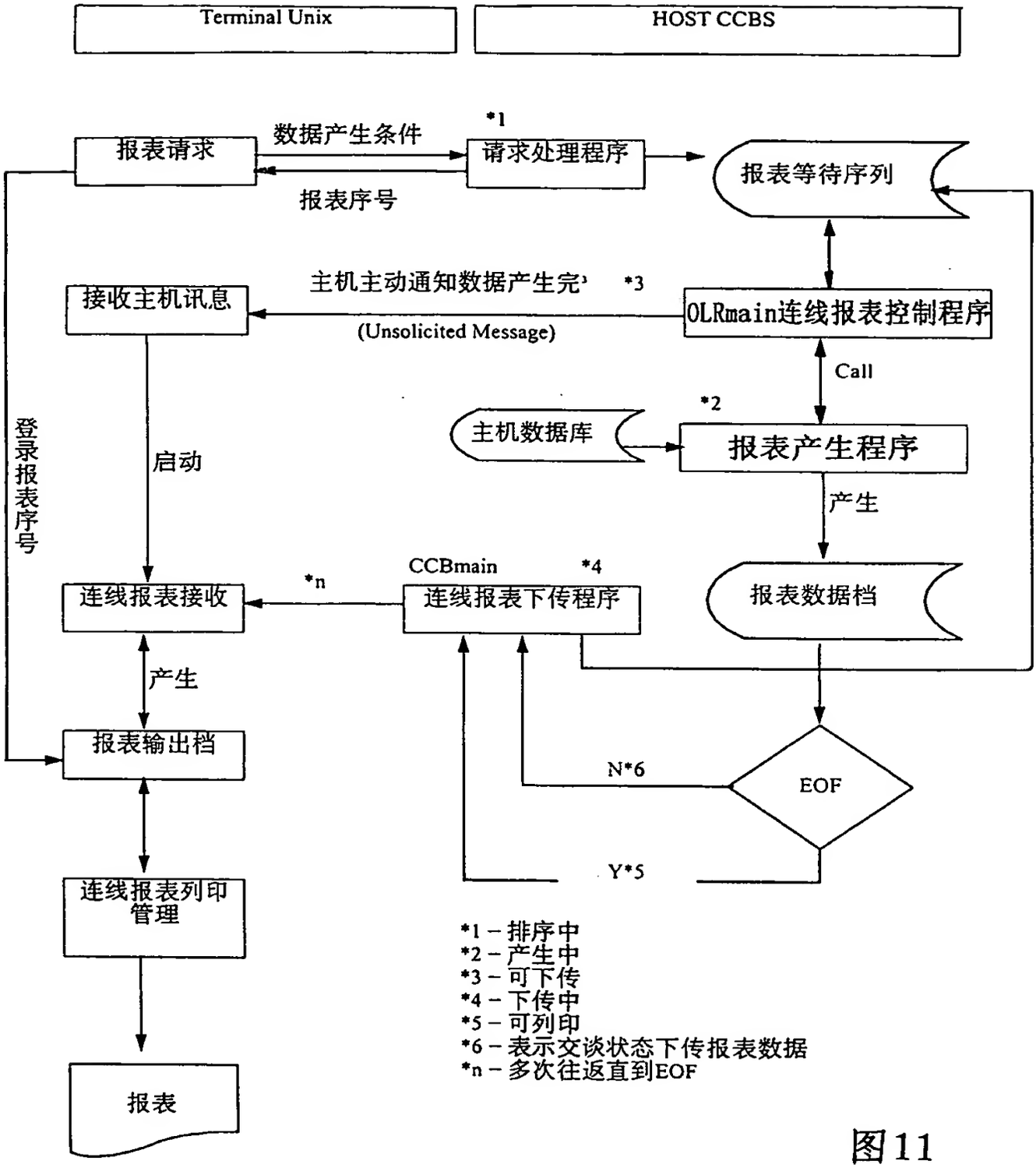

Terminal Unix
HOST CCBS
报表请求
数据产生条件
*1
请求处理程序
报表等待序列
报表序号
主机主动通知数据产生完
*3
接收主机讯息
(Unsolicited Message)
OLRmain连线报表控制程序
Call
*2
主机数据库
报表产生程序
登录报表序号
启动
产生
CCBmain
*4
*n
连线报表接收
连线报表下传程序
报表数据档
产生
报表输出档
N*6
EOF
Y*5
连线报表列印管理
报表
*1 – 排序中
*2 – 产生中
*3 – 可下传
*4 – 下传中
*5 – 可列印
*6 – 表示交谈状态下传报表数据
*n – 多次往返直到EOF

图11

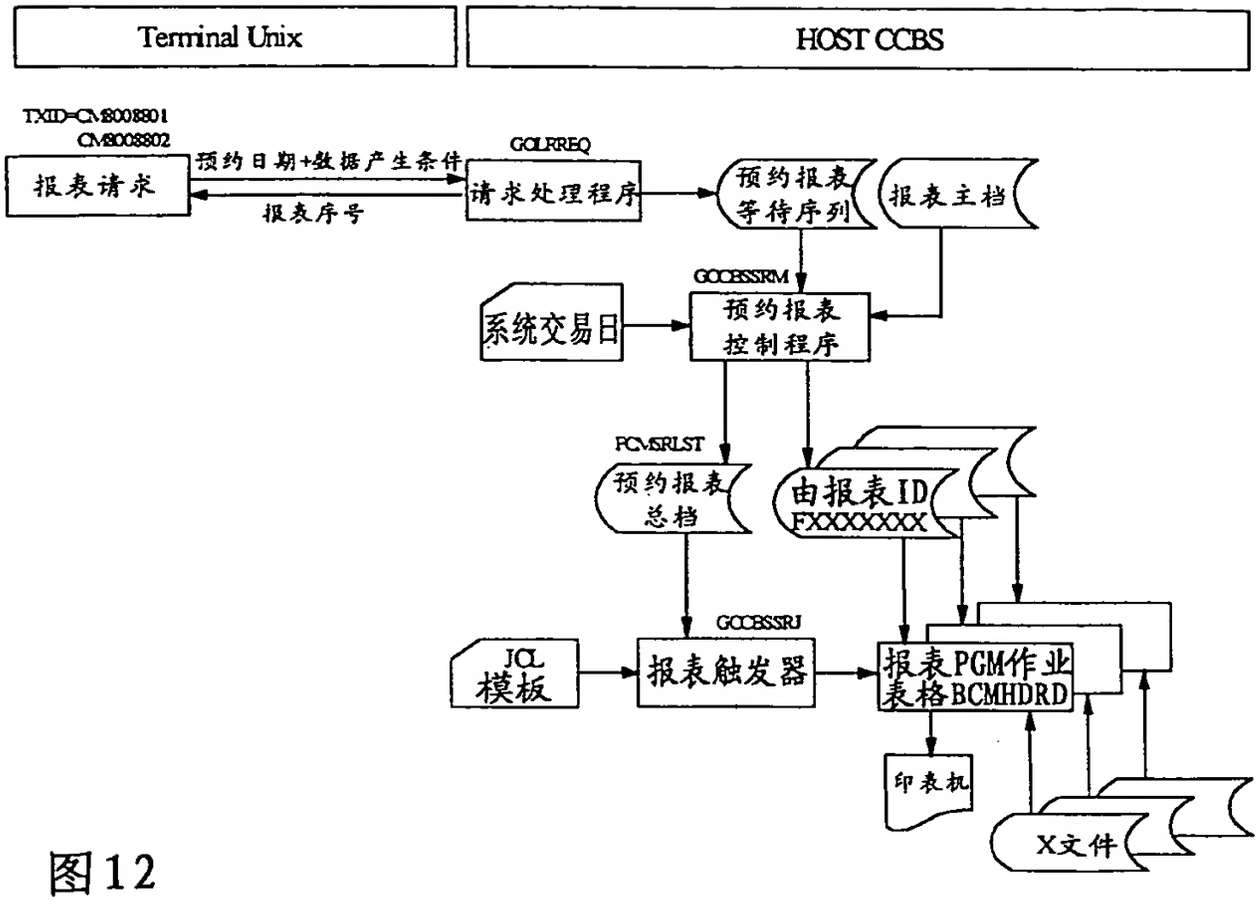

Terminal Unix
HOST CCBS
TXID=CM8008801
CM8008802
报表请求
预约日期+数据产生条件
报表序号
GOLFREQ
请求处理程序
预约报表
等待序列
报表主档
GOCBSSRM
系统交易日
预约报表
控制程序
FCMSRLST
预约报表
总档
由报表ID
FXXXXXXX
GOCBSSRJ
JCL
模板
报表触发器
报表PGM作业
表格BCMHDRD
印表机
X文件

图12

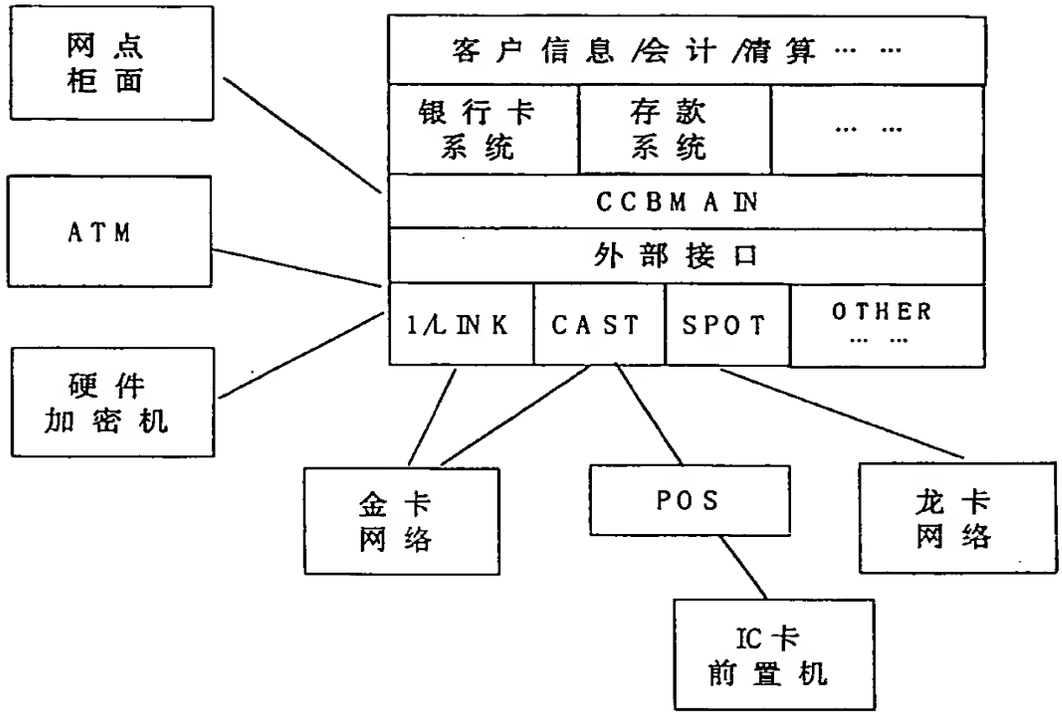

网点
柜面
ATM
硬件
加密机
客户信息/会计/清算……
银行卡
系统
存款
系统
……
CCBMAIN
外部接口
1ℓINK
CAST
SPOT
OTHER
……
金卡
网络
POS
龙卡
网络
IC卡
前置机

图13

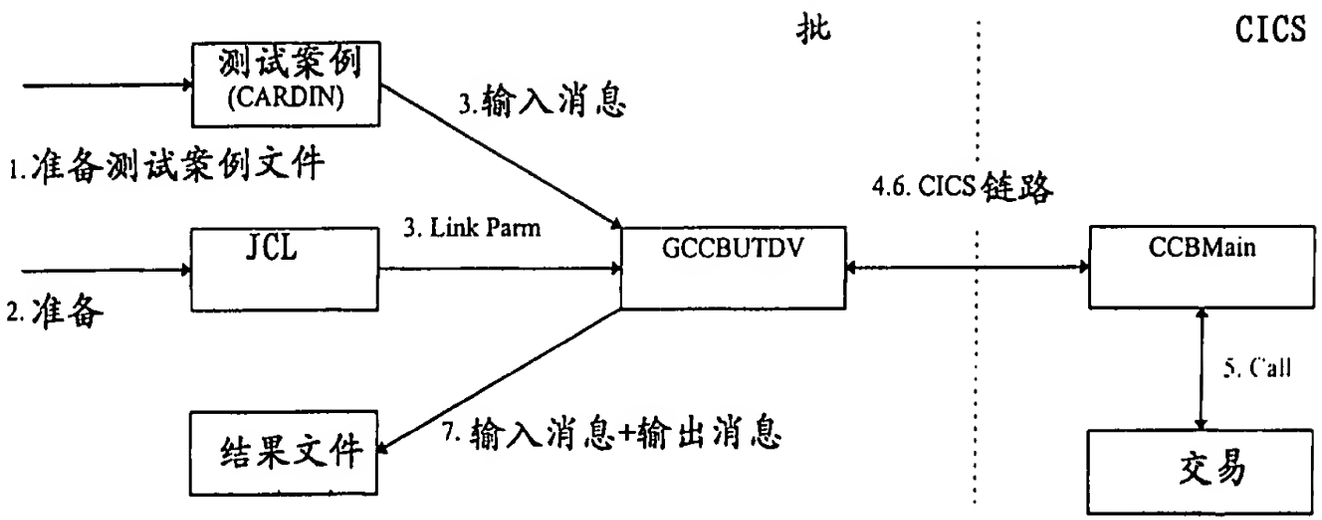

批
CICS
测试案例
(CARDIN)
3.输入消息
1.准备测试案例文件
4.6. CICS链路
JCL
3. Link Parm
GCCBUTDV
CCBMain
2.准备
5. Call
结果文件
7. 输入消息+输出消息
交易

图14

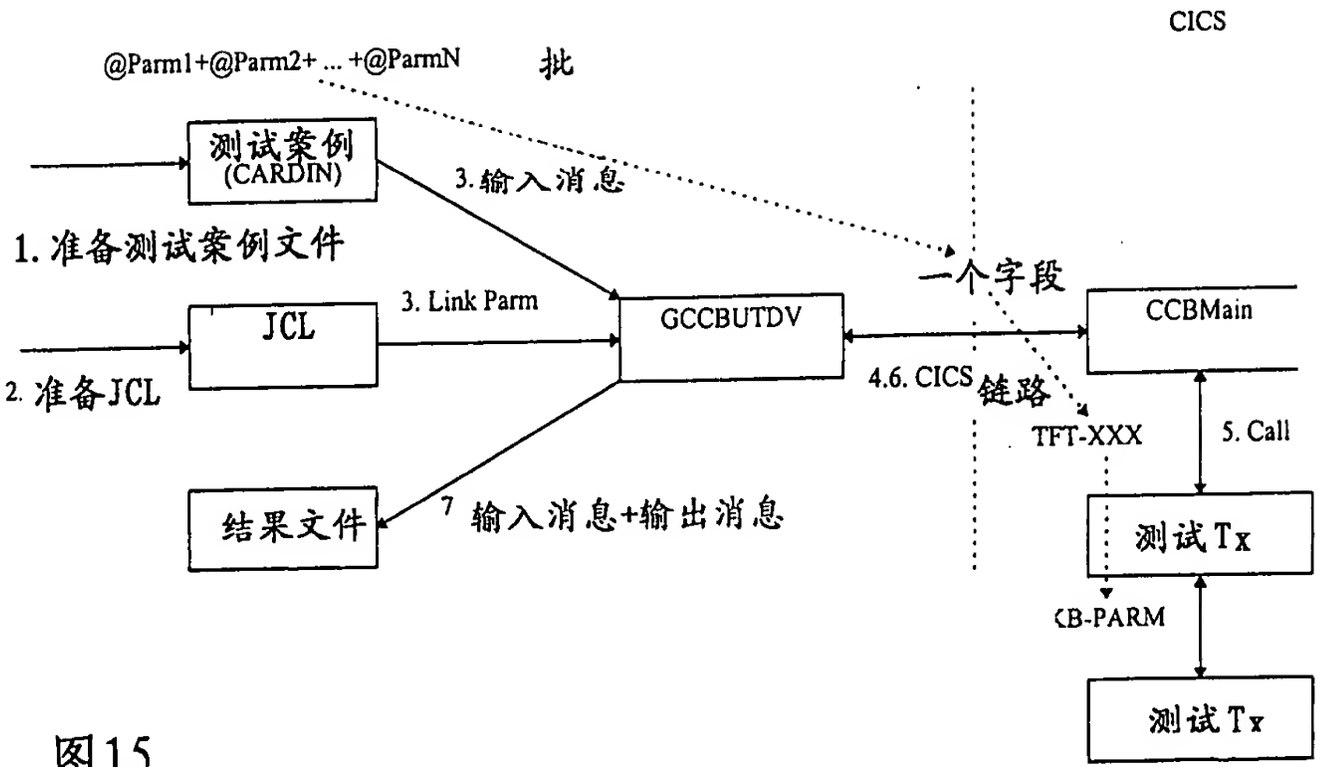

CICS
@Parm1+@Parm2+ ... +@ParmN
批
测试案例
(CARDIN)
3. 输入消息
1. 准备测试案例文件
一个字段
3. Link Parm
JCL
GCCBUTDV
CCBMain
2. 准备JCL
4.6. CICS 链路
TFT-XXX
5. Call
结果文件
7 输入消息+输出消息
测试Tx
CB-PARM
测试Tx

图15

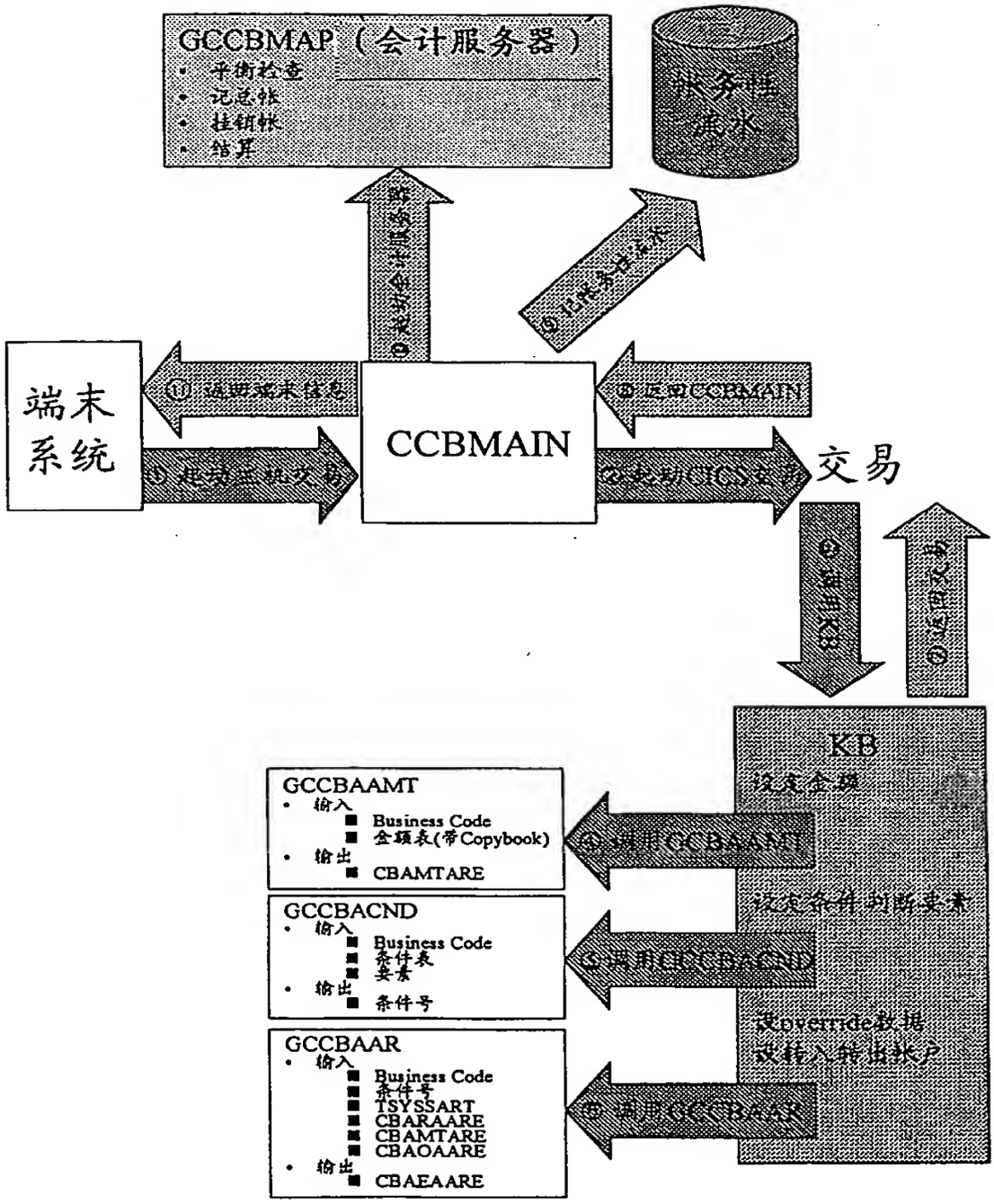

GCCBMAP（会计服务器）
• 平衡检查
• 记总帐
• 挂销帐
• 结算
端末系统
CCBMAIN
交易
KB
GCCBAAMT
• 输入
■ Business Code
■ 金额表(带Copybook)
• 输出
■ CBAMTARE
GCCBACND
• 输入
■ Business Code
■ 条件表
■ 要素
• 输出
■ 条件号
GCCBAAR
• 输入
■ Business Code
■ 条件号
■ TSYSSART
■ CBARAARE
■ CBAMTARE
■ CBAOAARE
• 输出
■ CBAEAARE

图16

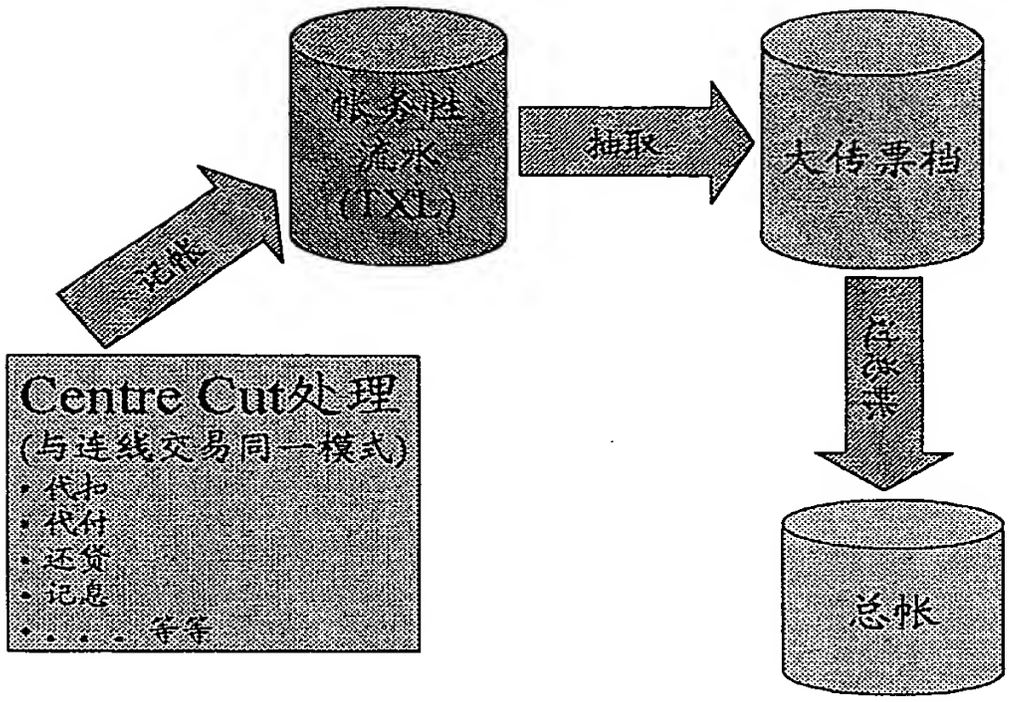

帐务性
流水
(TXL)
抽取
大传票档
记帐
Centre Cut处理
(与连线交易同一模式)
• 代扣
• 代付
• 还贷
• 记息
• • • 等等
存余额
总帐

图17

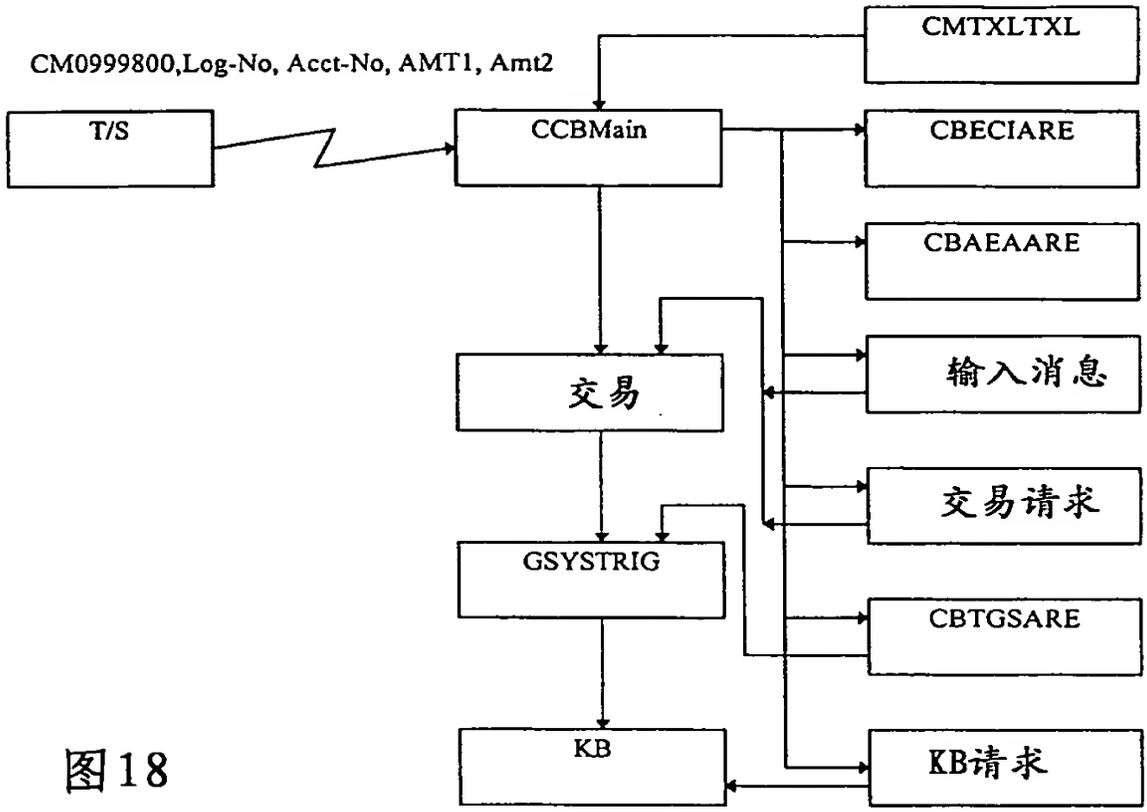

CM0999800,Log-No, Acct-No, AMT1, Amt2
T/S
CCBMain
CMTXLTXL
CBECIARE
CBAEAARE
交易
输入消息
交易请求
GSYSTRIG
CBTGSARE
KB
KB请求

图18

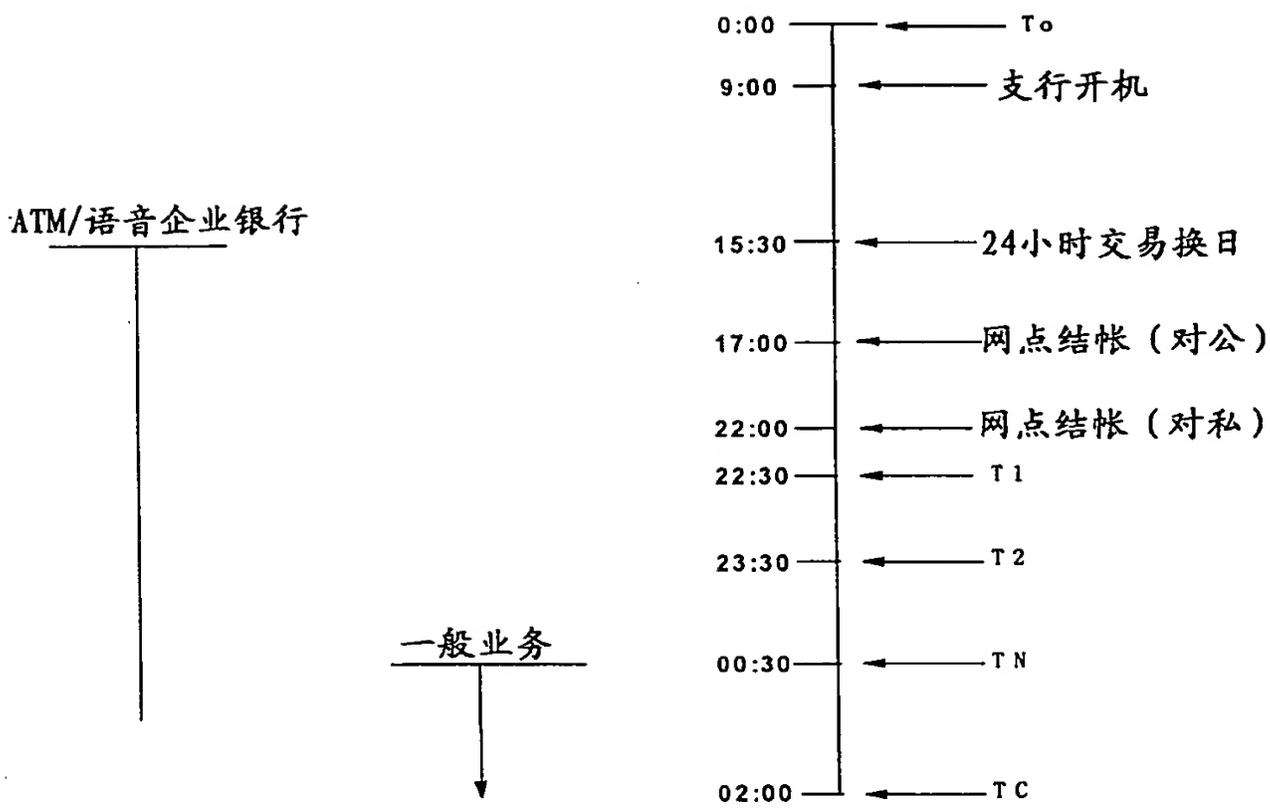

0:00
To
9:00
支行开机
ATM/语音企业银行
15:30
24小时交易换日
17:00
网点结帐（对公）
22:00
网点结帐（对私）
22:30
T1
23:30
T2
一般业务
00:30
TN
02:00
TC

图19

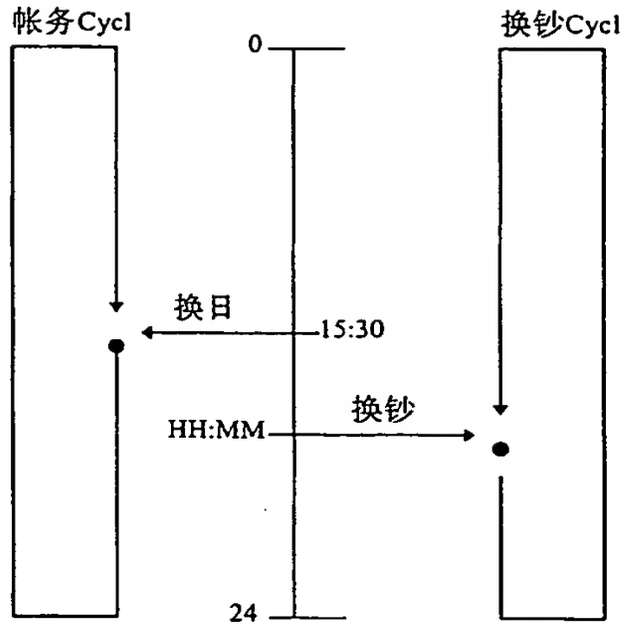
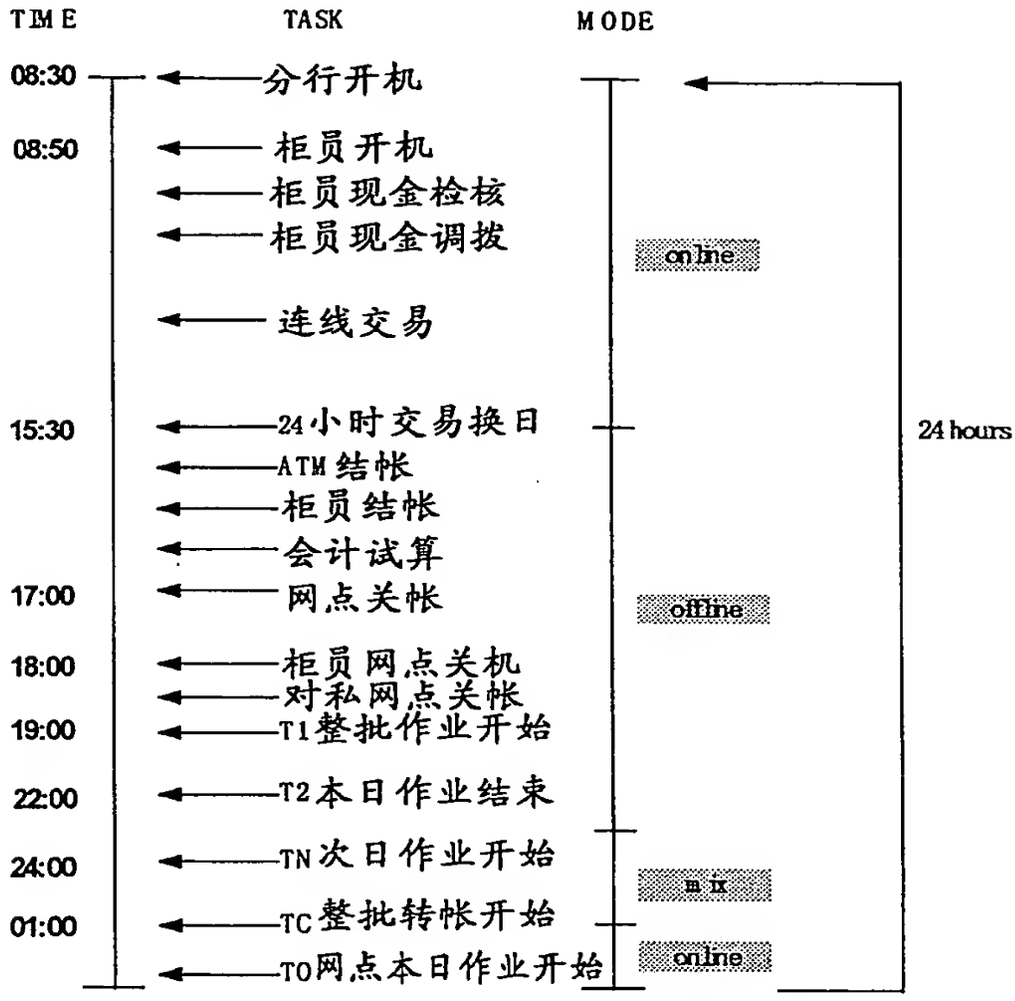


图20

ATM 作业模式与整批时段 Time Slot)



1. online同时记业务帐与会计帐。
2. offline记暂存档

3. mix依帐户状态记帐
4. 24小时业务比照ATM处理。

图21

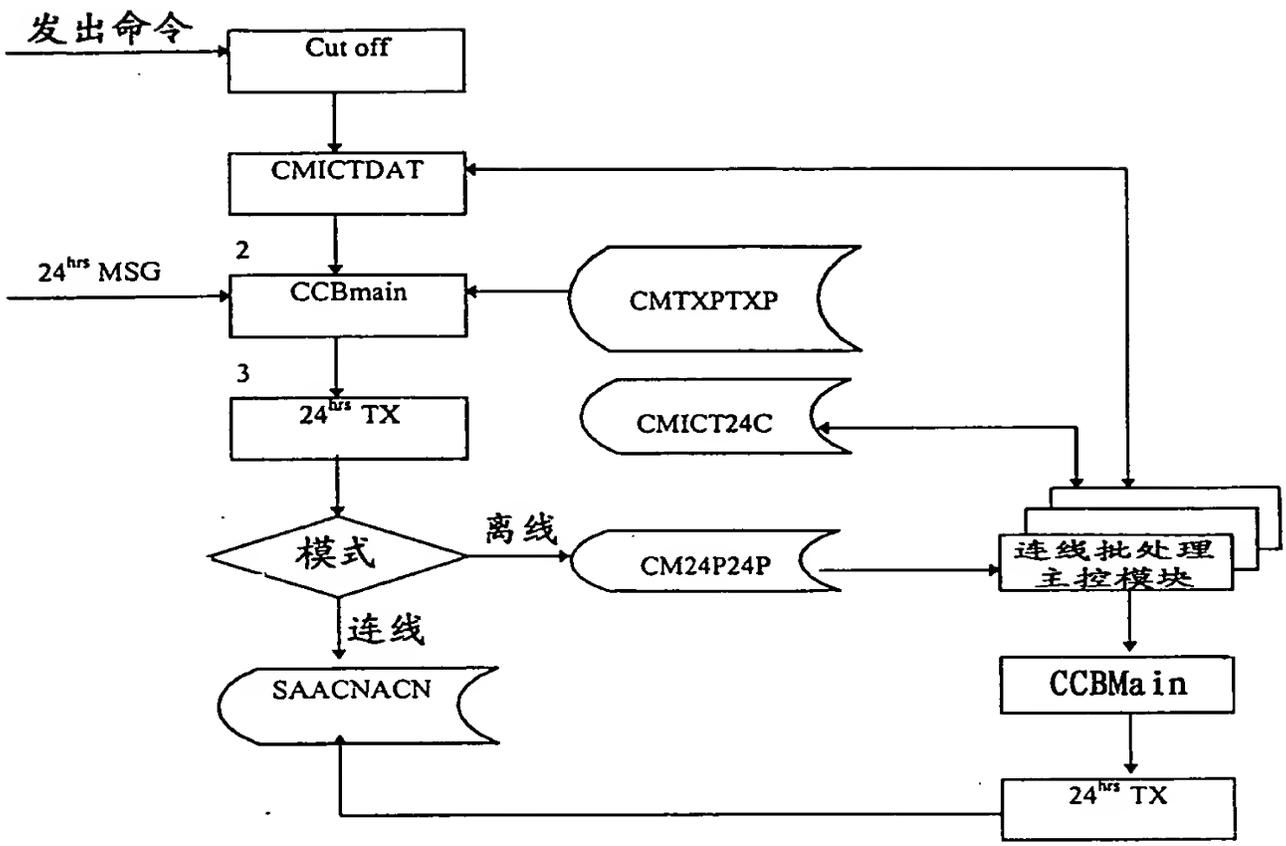

发出命令
Cut off
CMICTDAT
2
24hrs MSG
CCBmain
CMTXPTXP
3
24hrs TX
CMICT24C
模式
离线
CM24P24P
连线批处理
主控模块
连线
SAACNACN
CCBMain
24hrs TX

图22

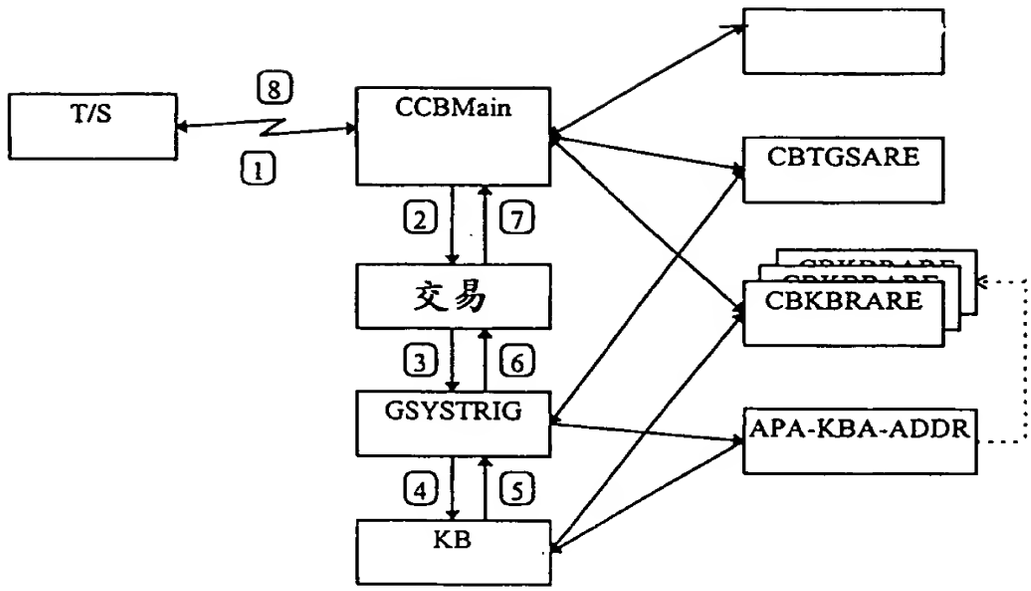

T/S
8
1
CCBMain
2
7
交易
3
6
GSYSTRIG
4
5
KB
CBTGSARE
CBKBRARE
APA-KBA-ADDR

图23